ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Υπολογιστικών Συστημάτων

Σύστημα Εντοπισμού και Tracking αντικειμένων για την καθοδήγηση UAV με τη συμβολή απομακρυσμένου server

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
του

Οδυσσέα Κ. Ντούση

Επιβλέπων: Παναγιώτης Δ. Τσανάκας
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2024

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Υπολογιστικών Συστημάτων

Σύστημα Εντοπισμού και Tracking αντικειμένων για την καθοδήγηση UAV με τη συμβολή απομακρυσμένου server

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
του

Οδυσσέα Κ. Ντούση

Επιβλέπων: Παναγιώτης Δ. Τσανάκας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 4/10/2024.

.....	.....	.....
Χρήστος Παυλάτος	**Παναγιώτης Τσανάκας**	**Δημήτριος Σούντρης**
Επ. Καθηγητής, Σχολή Ικάρων	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2024

.....
Οδυσσέας Κ. Ντούσης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ο στόχος της παρούσας διπλωματικής εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός συστήματος που θα επιτρέπει την αυτόματη πλοήγηση ενός UAV και θα του δίνει τη δυνατότητα να εκτελέσει μια ανατεθειμένη αποστολή χρησιμοποιώντας οπτικές πληροφορίες. Ένα απλό παράδειγμα αυτού του είδους εξετάζεται σε αυτή την περίπτωση. Συγκεκριμένα, η αποστολή που έχει ανατεθεί στο UAV είναι η ανίχνευση, tracking και ακολούθηση ενός υποδειγμένου αντικειμένου (οχήματος).

Για την επίτευξη αυτού, προτείνεται μια αρχιτεκτονική 2 σταδίων, με σκοπό την υπέρβαση του περιορισμού των ανεπαρκών υπολογιστικών πόρων στο UAV. Το πρώτο στάδιο βασίζεται σε μια φορητή μονάδα ικανή για ελαφριές υπολογιστικές διεργασίες (ένα Raspberry Pi) που βρίσκεται στο UAV, μαζί με επιταχυντικό υλικό που μπορεί να υπολογίσει τα αποτελέσματα εκτέλεσης ελαφριών νευρωνικών δικτύων (Intel NCS2). Αυτό το υποσύστημα είναι υπεύθυνο για την ανίχνευση αντικειμένων, την παρακολούθηση του συγκεκριμένου στόχου με έναν απλό ανιχνευτή (tracker) και τη δημιουργία εντολών πλοήγησης. Το δεύτερο στάδιο είναι ένα απομακρυσμένο σύστημα που λειτουργεί σε έναν διακομιστή υψηλών επιδόσεων, στον οποίο συνδέεται το φορητό σύστημα. Ο διακομιστής εκτελεί πιο αξιόπιστη ανίχνευση αντικειμένων και παρακολουθεί τα αντικείμενα αυτά με έναν tracker βαθιάς μάθησης. Επίσης, προβλέπει τις μελλοντικές θέσεις του στόχου και εξάγει ορισμένα χαρακτηριστικά του από τα μοντέλα ανίχνευσης. Χρησιμοποιώντας αυτά τα χαρακτηριστικά και τις προβλεπόμενες θέσεις, επαναεντοπίζει τον στόχο σε περίπτωση απώλειάς του και μεταδίδει τις κατάλληλες οδηγίες στο UAV.

Το μοντέλο ανίχνευσης που χρησιμοποιείται στο Raspberry Pi είναι το YOLOv5n, δεδομένου ότι η υποστήριξη του NCS 2 έχει διακοπεί και οι επιλογές ήταν περιορισμένες, ενώ χρησιμοποιήθηκε ένας ελαφρύς ανιχνευτής υλοποιημένος στην OpenCV (KCF) για τη διαδικασία παρακολούθησης. Για τον διακομιστή, δοκιμάστηκαν τόσο μοντέλα βασισμένα σε συνελικτικά νευρωνικά δίκτυα όσο και σε transformers, με τα YOLOv8x και RT-DETR να επιτυγχάνουν τα καλύτερα (παρόμοια) αποτελέσματα. Τελικά, επιλέχθηκε το YOLOv8 λόγω πλεονεκτήματός του στον χρόνο συμπερασμού. Τόσο για το διακομιστή οσο και για το Raspberry Pi, τα μοντέλα ανίχνευσης εκπαιδεύτηκαν σε ένα κατάλληλο σύνολο δεδομένων, λόγω της ιδιαίτερης φύσης των δεδομένων που πρέπει να επεξεργαστούν. Η διαδικασία παρακολούθησης εκτελέστηκε χρησιμοποιώντας τον ανιχνευτή DeepSORT, ενώ ένα δίκτυο LSTM χρησιμοποιήθηκε για την πρόβλεψη της μελλοντικής τοποθεσίας του στόχου.

Λέξεις Κλειδιά: Κατανεμημένο Σύστημα, Καθοδήγηση UAV, Παρακολούθηση Αντικειμένων, Ανίχνευση Αντικειμένων, Βαθιά Νευρωνικά Δίκτυα, Δίκτυα LSTM, Νευρωνικά Δίκτυα Transformers, Πρόβλεψη Τροχιάς, Επαναεντοπισμός Στόχου, Αυτόνομη Πλοήγηση, Φορητή Υπολογιστική

Abstract

The objective of the present thesis is the design and implementation of a system that can allow the automatic navigation of a UAV and enable it to perform an assigned mission using visual information. A simple example of this sort is examined in this case. Specifically, the task assigned to the UAV is the detection, tracking and following of an indicated object (vehicle).

To achieve this, a 2 stage architecture is proposed, in order to overcome the limitation of inadequate computational resources on the UAV. The first stage is based on a portable unit capable of light computations (a Raspberry Pi) located on the UAV, along with accelerating hardware that can perform inference with light neural networks (Intel NCS2). This subsystem is responsible for detecting objects, tracking the specified target with a simple tracker, and generating navigation commands. The second stage is a remote system operating on a high performance server, to which the portable system is connected. The server performs more reliable object detection and tracks objects with a deep learning tracker. It also predicts the future positions of the target and extracts its characteristic features from the detection models. Using these features and the predicted locations, it relocates the target in case it is lost, and transmits appropriate directions to the UAV.

The detection model used on the Raspberry Pi is a YOLOv5n model, since the NCS 2 is discontinued and the options were limited, and a light opencv tracker (KCF) was used for the tracking process. For the server, both convolutional and transformer-based models were tested, with YOLOv8x and RT-DETR achieving the best (similar) results. YOLOv8 was eventually selected due to its inference time advantage. For both the server and the Raspberry Pi, the detection models were trained on an appropriate dataset, due to the uncommon nature of the data to be processed. The tracking process was performed using the DeepSORT tracker, and an LSTM network was utilized for the mentioned location prediction.

Key Words: Distributed System, UAV Guidance, Object Tracking, Object Detection, Deep Neural Networks, LSTM Networks, Transformer-based Neural Networks, Trajectory Prediction, Target Relocation, Autonomous Navigation, On-board Computing

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

1.1 Περιγραφή Αντικειμένου

Η ταχεία πρόοδος στον τομέα της Τεχνητής Νοημοσύνης τα τελευταία χρόνια έχει επιτρέψει τη χρήση μοντέλων Μηχανικής Μάθησης για ολοένα και πιο σύνθετες εργασίες, οι οποίες συχνά είναι κρίσιμης σημασίας. Μία τέτοια εφαρμογή είναι η αυτόνομη πλοήγηση Μη Επανδρωμένων Αεροχημάτων σε πολύπλοκες καταστάσεις, με στόχο την επιτυχή ολοκλήρωση μιας ανατεθειμένης αποστολής.

Ο στόχος αυτής της διπλωματικής εργασίας είναι η ανάπτυξη ενός συστήματος ικανού να πλοηγεί αυτόματα ένα UAV με αποστολή την παρακολούθηση (tracking) και ακολούθηση ενός συγκεκριμένου στόχου (οχήματος). Ο σκοπός είναι να εξοπλιστεί το UAV με την ικανότητα να επεξεργάζεται τα δεδομένα εισόδου του (τη ροή της κάμεράς του) με σύγχρονα μοντέλα Μηχανικής Μάθησης, ώστε να λαμβάνει πληροφορίες για τον στόχο και να παίρνει αποφάσεις πλοήγησης με βάση πιο αξιόπιστα δεδομένα.

1.2 Προκλήσεις

Η πιο σημαντική και προφανής πρόκληση που θα παρουσιαστεί είναι η έλλειψη των απαραίτητων υπολογιστικών πόρων στο UAV. Τα μοντέλα Μηχανικής Μάθησης που θα χρειαστεί να χρησιμοποιήσουμε απαιτούν σημαντική επεξεργαστική ισχύ, είτε αυτή αφορά την εξαγωγή χαρακτηριστικών με χρήση συνελικτικών φίλτρων, όπως στα μοντέλα YOLO, είτε τον υπολογισμό των τιμών προσοχής πολλαπλών κεφαλών στα μοντέλα τύπου transformer όπως ο DETR. Αυτό καθιστά αδύνατη τη χρήση των περισσοτέρων μοντέλων στο όχημα, λαμβάνοντας υπόψη ότι το επεξεργαστικό κέντρο του οχήματος θα είναι ένα Raspberry Pi.

Δεδομένου ότι αυτό είναι ένα πρόβλημα που δεν μπορεί εύκολα να λυθεί με την απλή προσθήκη εξοπλισμού στο UAV, ελαφρύτερη επεξεργασία θα πραγματοποιείται στο όχημα και τα δεδομένα θα μεταδίδονται σε έναν απομακρυσμένο server, ο οποίος δεν θα υπόκειται σε αυτούς τους περιορισμούς. Αυτό, ωστόσο, δημιουργεί τη νέα πρόκληση της βέλτιστης ανάθεσης εργασιών και συνδυασμού των παραγόμενων αποτελεσμάτων. Επίσης, εισάγει το πρόβλημα της μετάδοσης δεδομένων μέσω δικτύου κινητής τηλεφωνίας, σε περίπτωση απουσίας σήματος

WiFi, κάτι που με τη σειρά του προκαλεί προβληματισμό σχετικά με την ποιότητα της σύνδεσης και την ενδεχομένως μειωμένη ταχύτητα επικοινωνίας.

Είναι επίσης προφανές ότι η επεξεργασία των δεδομένων σε περίπου πραγματικό χρόνο είναι κρίσιμη. Σημαντικές καθυστερήσεις στη λήψη αποφάσεων (που μπορεί να προκληθούν σε οποιοδήποτε μέρος του συστήματος) θα οδηγήσουν πιθανότατα στην οριστική απώλεια του στόχου ή στην ανάγκη συνεχούς επαναεντοπισμού, κάτι που θα μειώσει σημαντικά την αξιοπιστία.

Υπάρχουν επίσης σημαντικές δυσκολίες που αφορούν τα ίδια τα μοντέλα. Πρώτον, λόγω της ασυνήθιστης γωνίας των καρέ, δεν θα είναι δυνατόν να χρησιμοποιηθούν προεκπαιδευμένα μοντέλα, που μπορεί να δυσκολευτούν να αναγνωρίσουν σωστά τα οχήματα ενδιαφέροντος. Επιπλέον, δεδομένου ότι ο στόχος θα είναι ένα συγκεκριμένο όχημα, πιθανόν να χρειαστεί να γίνει επαναταυτοποίηση, κάτι που μπορεί να αποδειχθεί δύσκολο λόγω της συνεχούς κίνησης του UAV και της κίνησης οχημάτων καθώς και της ενδεχόμενης αλλαγής γωνίας, που μπορεί να αλλοιώσει τα χαρακτηριστικά του στόχου. Θα χρειαστεί επίσης να αντιμετωπιστούν κοινά προβλήματα της ανίχνευσης και παρακολούθησης αντικειμένων, όπως προβληματικές εικόνες (π.χ. σκοτάδι, ομίχλη), μερικές ή πλήρεις αποκρύψεις αντικειμένων, πολλαπλές ανιχνεύσεις κ.λπ.

1.3 Δομή Κεφαλαίων

Στο πρώτο κεφάλαιο περιλαμβάνεται αυτή η σύντομη εισαγωγή, με περιγραφή του αντικειμένου και των προκλήσεων που θα πρέπει να επιλυθούν, καθώς και μια επισκόπηση της δομής της αναφοράς. Στο δεύτερο κεφάλαιο ακολουθεί μια παρουσίαση βιβλιογραφίας σχετικής με το υπό μελέτη αντικείμενο, μαζί με μια απόπειρα επαρκούς ανάλυσης των μοντέλων που πρόκειται να δοκιμαστούν. Στο τρίτο κεφάλαιο εξετάζονται τα διαθέσιμα δεδομένα εκπαίδευσης και οι πληροφορίες που παρέχουν μέσα από τα δείγματα και τις επισημάνσεις τους. Συζητούνται επίσης οι απαραίτητες μορφές δεδομένων και η προεπεξεργασία που χρειάστηκε να γίνει. Στη συνέχεια, γίνεται μια περιγραφή υψηλού επιπέδου του προτεινόμενου μοντέλου και των κύριων συνιστωσών του συστήματος (τόσο σε επίπεδο υλικού όσο και λογισμικού). Στο κεφάλαιο 5, υπάρχει αναλυτική περιγραφή του λογισμικού που εκτελείται στο UAV, συμπεριλαμβανομένου του μοντέλου ανίχνευσης, του tracker, της διαδικασίας μετάδοσης δεδομένων (στις περιπτώσεις παρουσίας ή απουσίας σύνδεσης WiFi) και τέλος της δημιουργίας των εντολών που θα κατευθύνουν το όχημα. Στο κεφάλαιο 6, υπάρχει αναλυτική περιγραφή των δοκιμών που πραγματοποιήθηκαν για διάφορα μοντέλα ανίχνευσης αντικειμένων και πρόβλεψης τροχιάς που θα εκτελούνται στον απομακρυσμένο server που θα χρησιμοποιηθεί, καθώς και η διαδικασία εκπαίδευσης που ακολουθήθηκε για την προσαρμογή αυτών των μοντέλων σε εξειδικευμένα δεδομένα, ώστε να βελτιωθούν τα παραγόμενα αποτελέσματα. Παρέχεται επίσης περιγραφή της μεθόδου που χρησιμοποιήθηκε για την επαναεντοπισμό του στόχου όταν χρειάζεται. Τέλος, το κεφάλαιο 7 περιλαμβάνει ορισμένα γενικά συμπεράσματα και κατευθύνσεις για μελλοντική εργασία.

Κεφάλαιο 2

Σχετική Βιβλιογραφία και Υπόβαθρο

2.1 Ανίχνευση Αντικειμένων

Η ανίχνευση αντικειμένων σε εικόνες ήταν πάντα μία από τις πιο σημαντικές προκλήσεις της όρασης υπολογιστών. Οι πρώτες προσεγγίσεις βασίζονταν σε παραδοσιακές τεχνικές και περιορισμένους υπολογιστικούς πόρους, οδηγώντας στην ανάπτυξη ανιχνευτών όπως ο HOG [8] ή ο Ανιχνευτής Viola-Jones [38]. Με την εξέλιξη των νευρωνικών δικτύων και τη βελτίωση των διαθέσιμων υπολογιστικών συστημάτων, πιο σύνθετα μοντέλα μηχανικής μάθησης κυριάρχησαν στον τομέα: Αρχικά ανιχνευτές βασισμένοι σε Συνελικτικά Νευρωνικά Δίκτυα (CNNs) και πιο πρόσφατα μοντέλα βασισμένα σε Transformers.

2.1.1 Συνελικτικά Μοντέλα

Τα Συνελικτικά Νευρωνικά Δίκτυα χρησιμοποιούνται για την εξαγωγή χαρακτηριστικών από πίνακες συνδυάζοντας πληροφορίες από "γειτονιές" του πίνακα εισόδου, παράγοντας έναν νέο πίνακα (χάρτη χαρακτηριστικών). Αυτός ο συνδυασμός πραγματοποιείται με τη συνέλιξη της εισόδου με ένα σύνολο εκπαιδευμένων φίλτρων.

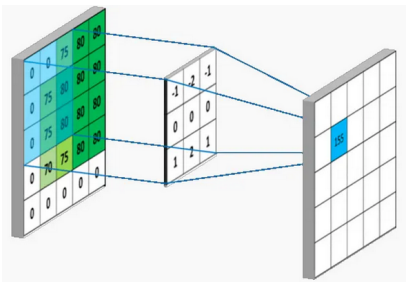


Figure 2.1: Υπολογισμός ενός στοιχείου του χάρτη χαρακτηριστικών που αντιστοιχεί σε μια περιοχή της εισόδου (Εικόνα απο towardsdatascience.com)

Αυτή η διαδικασία μπορεί να επαναληφθεί για τους παραγόμενους χάρτες χαρακτηριστικών μετά το pooling, για την εξαγωγή χαρακτηριστικών από μεγαλύτερες περιοχές της εισόδου. Τα pooling layers επιλέγουν αντιπροσωπευτικές τιμές (για παράδειγμα μέγιστη ή μέση τιμή) για τμήματα των χαρτών χαρακτηριστικών, μειώνοντας το μέγεθός τους. Μια συνάρτηση ενεργοποίησης όπως η RELU (RELU(x)=max(x,0)) μπορεί να χρησιμοποιηθεί μετά από κάθε συνελικτικό επίπεδο πριν από το pooling. Η τελική έξοδος του δικτύου (για παράδειγμα η ταξινόμηση εικόνας) παράγεται τροφοδοτώντας τον τελευταίο χάρτη χαρακτηριστικών σε ένα πλήρως συνδεδεμένο επίπεδο.

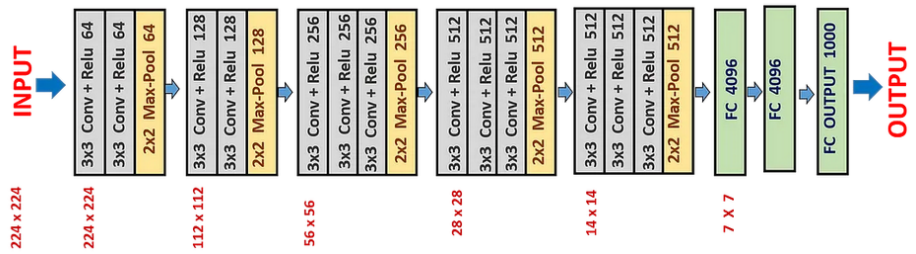


Figure 2.2: Παράδειγμα της δομής ενός CNN (VGGNet-16) (Εικόνα απο medium.com)

Οι συνελικτικοί ανιχνευτές βασίζονται στην εξαγωγή χαρακτηριστικών από την εικόνα χρησιμοποιώντας CNNs, και στη χρήση αυτών των πληροφοριών για τον εντοπισμό περιοχών της εικόνας που αντιστοιχούν σε αντικείμενα συγκεκριμένων κατηγοριών. Αυτοί μπορούν να ταξινομηθούν σε δύο κύριες κατηγορίες: ανιχνευτές ενός και δύο σταδίων.

Οι ανιχνευτές δύο σταδίων αρχικά δημιουργούν προτάσεις περιοχών, και σύμφωνα με αυτές τα αντίστοιχα τμήματα της εικόνας ταξινομούνται και τα bounding boxes εντοπίζονται με μεγαλύτερη ακρίβεια. Οι πιο ευρέως χρησιμοποιούμενοι ανιχνευτές δύο σταδίων είναι αυτοί της οικογένειας R-CNN. Τα μοντέλα ενός σταδίου (όπως το Retinanet [25] ή τα YOLO) δεν χρησιμοποιούν προτάσεις περιοχών, αλλά προβλέπουν bounding boxes και πιθανότητες κατηγοριών για προκαθορισμένα τμήματα της εικόνας. Οι ανιχνευτές ενός σταδίου είναι σημαντικά ταχύτεροι, κάτι που τους καθιστά πιο κατάλληλους για χρήση σε πραγματικό χρόνο.

Faster R-CNN

Τα μοντέλα της οικογένειας R-CNN είναι ανιχνευτές δύο σταδίων. Το πρώτο τέτοιο μοντέλο, το R-CNN [11], αποτελείται από τέσσερα κύρια λειτουργικά μπλοκ:

- **Μονάδα Πρότασης Περιοχών (Region Proposal Module):** Χρησιμοποιώντας τη μέθοδο Selective Search [36], ορίζονται περιοχές της εικόνας που ενδεχομένως περιέχουν αντικείμενα (περίπου 2000 στο αρχικό μοντέλο).
- **Εξαγωγέας Χαρακτηριστικών (Feature Extractor):** Με τη χρήση ενός CNN, εξάγεται ένα διάνυσμα χαρακτηριστικών για κάθε μία από τις παραπάνω περιοχές. Η είσοδος του CNN που χρησιμοποιείται στον αρχικό ανιχνευτή έχει διάσταση 227x227,

επομένως οι προτεινόμενες περιοχές της εικόνας μετασχηματίζονται ώστε να προσαρμοστούν σε αυτές τις διαστάσεις. Τα παραγόμενα διανύσματα έχουν μήκος 4096.

- **Ταξινομητής (Classifier):** Τα διανύσματα χαρακτηριστικών που εξάγονται από τις προτεινόμενες περιοχές ταξινομούνται τελικά με τη βοήθεια SVMs [7].

- **Bounding Box Regressor:** Μετά την ταξινόμηση των προτεινόμενων περιοχών, ένα μοντέλο γραμμικής παλινδρόμησης προβλέπει τη χαρτογράφηση από το προτεινόμενο bounding box σε ένα πιο κατάλληλο, χρησιμοποιώντας τα χαρακτηριστικά που παράγονται από το CNN (από το επίπεδο pool5 για το μοντέλο VGG).

Ένα από τα κύρια προβλήματα αυτής της αρχιτεκτονικής είναι ο πολύ μεγάλος χρόνος επεξεργασίας που προκύπτει από την επαναλαμβανόμενη εκτέλεση της διαδικασίας ανίχνευσης για όλες τις προτεινόμενες περιοχές. Επίσης, η διαδικασία εκπαίδευσης είναι αργή και πρέπει να πραγματοποιείται σε τρία ξεχωριστά στάδια (για το CNN, τα SVMs και τους εκτιμητές των bounding box). Ως μια απόπειρα υπέρβασης αυτών των ζητημάτων, προτάθηκε το Fast R-CNN [10]. Το Fast R-CNN υπολογίζει τα χαρακτηριστικά του CNN για ολόκληρη την εικόνα, και στη συνέχεια για κάθε προτεινόμενη περιοχή εξάγεται ένα διάνυσμα χαρακτηριστικών από το αποτέλεσμα. Αφού αυτό το διάνυσμα περάσει από μια ακολουθία πλήρως συνδεδεμένων επιπέδων, οι πιθανότητες κατηγοριοποίησης υπολογίζονται από ένα επίπεδο εξόδου και ένα βελτιωμένο bounding box παράγεται από έναν σχετικό regressor.

Ωστόσο, η μειωμένη ταχύτητα παρέμενε πρόβλημα, κυρίως λόγω της διαδικασίας πρότασης περιοχών. Για να αντιμετωπιστεί αυτό, το Faster R-CNN [34] εισήγαγε τα Δίκτυα Πρότασης Περιοχών (Region Proposal Networks - RPNs). Αφού ένας χάρτης χαρακτηριστικών για την αρχική εικόνα παραχθεί από ένα CNN εξαγωγής χαρακτηριστικών όπως στο Fast R-CNN, το RPN δημιουργεί προτάσεις περιοχών μετακινώντας ένα παράθυρο nxn (3x3 στην αρχική υλοποίηση) πάνω από αυτόν τον χάρτη χαρακτηριστικών. Κάθε ένα από αυτά τα παράθυρα αντιστοιχίζεται σε ένα διάνυσμα μήκους 256 ή 512 (ανάλογα με το βάθος του χάρτη χαρακτηριστικών), το οποίο περνάει από δύο πλήρως συνδεδεμένα επίπεδα: Ένα με 2 εξόδους, που εκτιμά την πιθανότητα ύπαρξης ή μη αντικειμένου, και ένα με 4 εξόδους, που εκτιμά τις συντεταγμένες του πιθανού bounding box. Για κάθε ένα από αυτά τα σημεία του χάρτη χαρακτηριστικών, το δίκτυο έχει σχεδιαστεί ώστε να κάνει προβλέψεις για k (k=9 στην αρχική υλοποίηση) anchor boxes διαφόρων διαστάσεων. Επομένως, οι τελικές έξοδοι των προαναφερθέντων πλήρως συνδεδεμένων επιπέδων είναι τελικά 2k και 4k για κάθε σημείο του χάρτη χαρακτηριστικών.

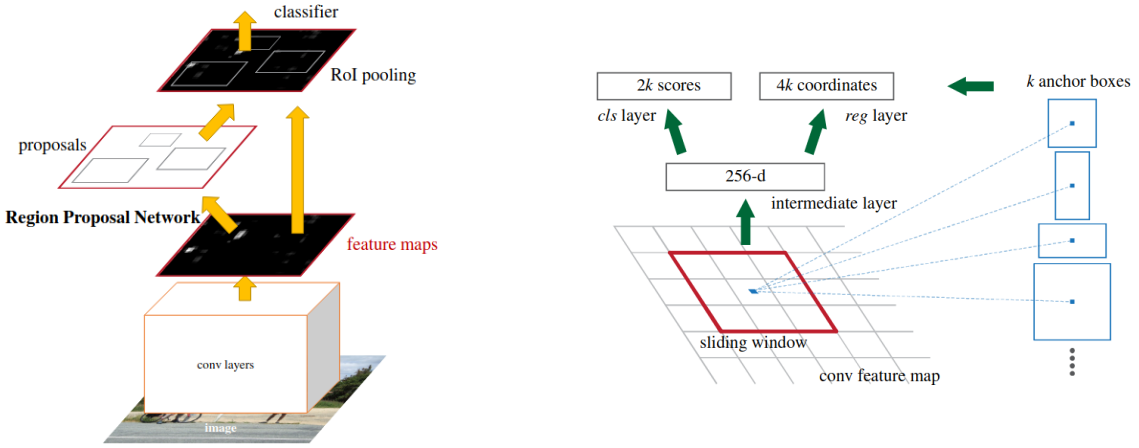


Figure 2.3: Στα αριστερά: Επισκόπηση της δομής του Faster R-CNN. Στα δεξιά: Δίκτυο Πρότασης Περιοχών. (Εικόνες απο [34])

Retinanet

Το Retinanet [25] είναι ο πρώτος ανιχνευτής ενός σταδίου που πέτυχε αποτελέσματα μέσης ακρίβειας στο σύνολο δεδομένων COCO [26] τα οποία ξεπερνούσαν εκείνα των τότε κορυφαίων ανιχνευτών δύο σταδίων. Αυτό επιτεύχθηκε με την εισαγωγή της Focal Loss, ξεπερνώντας ένα από τα σημαντικά προβλήματα των ανιχνευτών ενός σταδίου, την αναποτελεσματική εκπαίδευση λόγω ανισορροπίας κλάσεων - οι ανιχνευτές ενός σταδίου εκπαιδεύονται σε δεκάδες χιλιάδες πλαίσια ανά εικόνα, από τα οποία ένα μικρό ποσοστό περιέχει αντικείμενα ενδιαφέροντος, οδηγώντας σε μεγάλο αριθμό "εύκολων αρνητικών" παραδειγμάτων που αξιολογούνται με τον ίδιο τρόπο όπως τα πολύ λιγότερα και πιο σημαντικά δυσκολότερα παραδείγματα.

Στην περίπτωση της δυαδικής ταξινόμησης, η Focal Loss μπορεί να εκφραστεί ως $FL(p_t) = -a_t \cdot (1-p_t)^{\gamma} \cdot log(p_t)$, όπου:

$$p_t = \begin{cases} p & \text{αν πραγματική κλάση} = 1 \\ 1-p & \text{αλλιώς} \end{cases}, a_t = \begin{cases} \alpha & \text{αν πραγματική κλάση} = 1 \\ 1-\alpha & \text{αλλιώς} \end{cases},$$

όπου p είναι η προβλεπόμενη πιθανότητα για την κλάση 1, α είναι ένας παράγοντας κλίμακας στο [0,1] και γ είναι μια ρυθμιζόμενη παράμετρος εστίασης.

Ο παράγοντας $(1-p_t)^{\gamma}$ διασφαλίζει ότι τα παραδείγματα που ταξινομούνται εύκολα ($p_t >> 0.5$) έχουν μειωμένη επίδραση στη διαδικασία εκπαίδευσης. Ο a_t είναι ένας παράγοντας βαρύτητας που συμβάλλει στη ρύθμιση της ανισορροπίας που προκαλείται από διαφορετικούς αριθμούς δειγμάτων ανά κλάση, ανεξάρτητα από την ευκολία της ταξινόμησης.

Το Retinanet λειτουργεί αρχικά εξάγοντας χαρακτηριστικά από την εικόνα με ένα CNN (ResNet [12]), και χρησιμοποιώντας τα εξαγόμενα χαρακτηριστικά δημιουργεί μια πυραμίδα χαρακτηριστικών με ένα Δίκτυο Πυραμίδας Χαρακτηριστικών (Feature Pyramid Network - FPN) [24]. Τα χαρακτηριστικά από την πυραμίδα περνούν στη συνέχεια μέσα από δύο

παράλληλα υποδίκτυα, ένα για την ταξινόμηση και ένα για την εκτίμηση των bounding boxes.

Το υποδίκτυο ταξινόμησης είναι ένα Πλήρως Συνελικτικό Δίκτυο, το οποίο εκπαιδεύεται χρησιμοποιώντας την Focal Loss. Για κάθε σημείο του εισαγόμενου χάρτη χαρακτηριστικών, παράγει ένα διάνυσμα μήκους K·A, όπου A είναι ο αριθμός των anchor boxes και K είναι ο αριθμός των κλάσεων. Ο εκτιμητής bounding box είναι ένα άλλο Πλήρως Συνελικτικό Δίκτυο, παρόμοιο με το υποδίκτυο ταξινόμησης, αλλά για κάθε σημείο του χάρτη χαρακτηριστικών παράγει 4·A εξόδους.

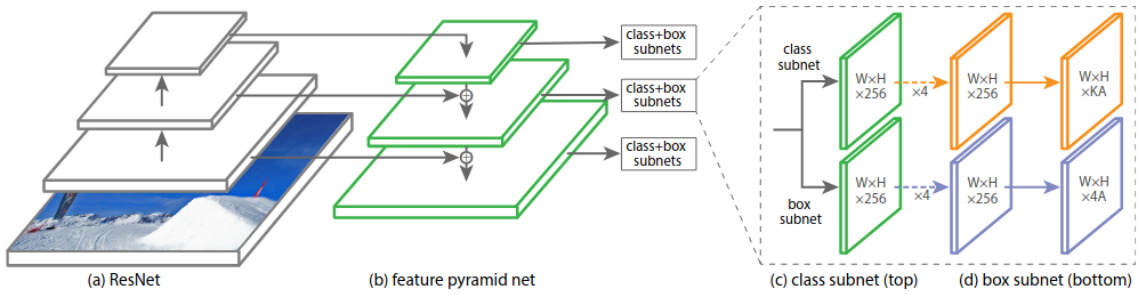


Figure 2.4: Επισκόπηση της αρχιτεκτονικής του ανιχνευτή Retinanet (Εικόνα απο [25])

Μοντέλα YOLO

Το πρώτο μοντέλο "You Only Look Once" που παρουσιάστηκε [33] είναι ένα συνελικτικό δίκτυο που επιχειρεί να προβλέψει τις συντεταγμένες των bounding boxes και τις πιθανότητες κλάσης ταυτόχρονα. Αποτελείται από 24 συνελικτικά επίπεδα και 2 πλήρως συνδεδεμένα επίπεδα που παράγουν τα τελικά αποτελέσματα.

Η εικόνα εισόδου χωρίζεται αρχικά σε ένα πλέγμα, με B πλαίσια να προβλέπονται για κάθε κελί του (B=2 στην αρχική υλοποίηση) και αντίστοιχα σκορ αξιοπιστίας. Η αξιοπιστία ορίζεται ως το γινόμενο της προβλεπόμενης πιθανότητας ύπαρξης αντικειμένου με την Τομή προς Ένωση (Intersection over Union) μεταξύ του προβλεπόμενου και του πραγματικού πλαισίου. Για κάθε κελί, επίσης προβλέπονται C πιθανότητες κλάσης. Το μοντέλο εκπαιδεύεται με μια συνάρτηση απώλειας που λαμβάνει υπόψη το σφάλμα ταξινόμησης για ένα κελί μόνο εάν αυτό το κελί αντιστοιχεί σε αντικείμενο, και το σφάλμα του bounding box εξετάζεται μόνο για την εκτίμηση του κελιού που ταιριάζει καλύτερα στο πραγματικό πλαίσιο (έχει το υψηλότερο IoU).

Λόγω της προαναφερθείσας απλής αρχιτεκτονικής και προσέγγισης του προβλήματος, το YOLO είχε σημαντικό πλεονέκτημα ταχύτητας σε σύγκριση με άλλους ανιχνευτές, επιτυγχάνοντας δυνατότητα λειτουργίας σε πραγματικό χρόνο - αν και αυτή η πρώτη έκδοση πέτυχε χαμηλότερη Μέση Ακρίβεια (Average Precision) σε σύγκριση με ανιχνευτές όπως το Faster R-CNN - και λαμβάνει υπόψη τις πληροφορίες περιβάλλοντος από ολόκληρη την εικόνα για κάθε πρόβλεψη. Επίσης, μαθαίνει χαρακτηριστικά αντικειμένων που είναι λιγότερο εξειδικευμένα σε μεμονωμένες περιπτώσεις, επιτρέποντας καλύτερη απόδοση σε νέους τύπους εισόδου.

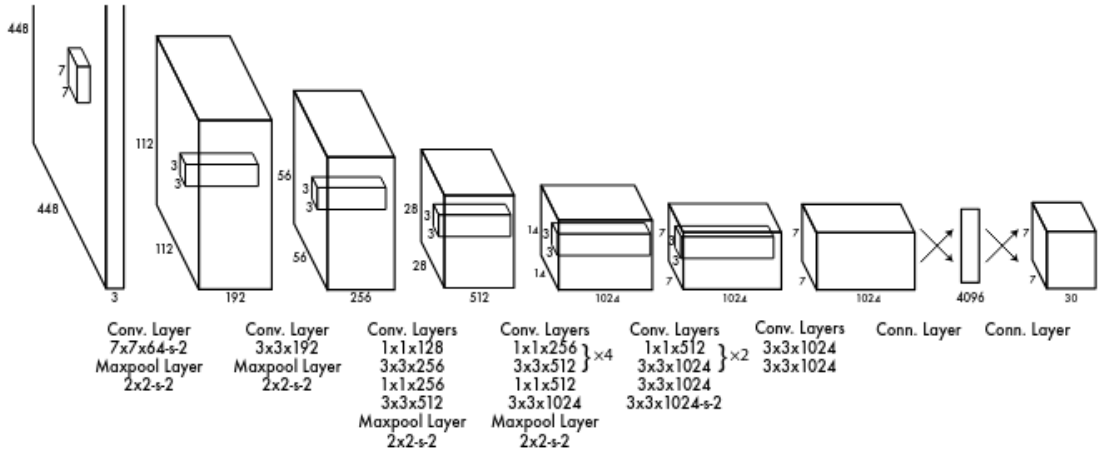


Figure 2.5: Επισκόπηση της αρχιτεκτονικής του πρώτου προτεινόμενου ανιχνευτή YOLO. Αυτό το μοντέλο έχει σχήμα πλέγματος 7x7, 20 κλάσεις και 2 προβλεπόμενα πλαίσια ανά κελί του πλέγματος, επομένως η τελική πρόβλεψη είναι ένα μπλοκ 7x7x30. (Εικόνα από [33])

Αυτός ο πρώτος ανιχνευτής YOLO αποτέλεσε τη βάση πάνω στην οποία έγιναν σημαντικές βελτιώσεις στις επόμενες εκδόσεις (η τελευταία είναι η YOLOv10), οδηγώντας αυτή τη μέθοδο στην επίτευξη αποτελεσμάτων αιχμής όσον αφορά την ταχύτητα και τη Μέση Ακρίβεια. Μερικές από τις πιο σημαντικές από αυτές τις βελτιώσεις σε κάθε έκδοση μπορούν να φανούν στον παρακάτω πίνακα, ενώ λεπτομέρειες μπορούν να βρεθούν στις αντίστοιχες αναφορές.

Μοντέλο	Σημαντικές Μετατροπές
YOLOv2 [31]	Δίκτυο Darknet-19, Anchor Boxes, k-means στα bounding boxes εκπαίδευσης για τον καθορισμό αρχικών συντεταγμένων, Κανονικοποίηση παρτίδας (Batch Normalization), Προσθήκη επιπέδου παράκαμψης (passthrough layer) για την ανίχνευση πιο λεπτομερών χαρακτηριστικών, Εκπαίδευση πολλαπλών κλιμάκων, Απευθείας πρόβλεψη θέσης bounding boxes.
YOLOv3 [32]	Δίκτυο Darknet-53, Προβλέψεις σε 3 κλίμακες με μηχανισμό παρόμοιο με το FPN [24], Πολλαπλή ταξινόμηση.
YOLOv4 [3]	Δίκτυο CSPDarknet53 [42], μπλόκ SPP [14], δίκτυο PANet [27]
YOLOv5 [19]	Δομή SPPF, Ενημερωμένος τύπος πρόβλεψης συντεταγμένων πλαισίων, απώλεια εκπαίδευσης ως σταθμισμένο άθροισμα των Απωλειών Ταξινόμησης (BCE Loss), Απώλεια ύπαρξης αντικειμένου (Objectness Loss) (BCE Loss) και Απώλεια Θέσης (Location Loss) (CIoU Loss)

Μοντέλο	Σημαντικές Μετατροπές
YOLOX [9]	Αποσυνδεδεμένη κεφαλή για ξεχωριστή ταξινόμηση, τοποθέτηση bounding boxes και πρόβλεψη ύπαρξης αντικειμένου, Ανίχνευση χωρίς anchor boxes.
YOLOv7 [40]	backbone E-ELAN , κύρια + βοηθητική κεφαλή για έξοδο και βαθιά εποπτευόμενη εκπαίδευση αντίστοιχα, προγραμματισμένη επαναπαραμετροποίηση
YOLOv6 [23]	backbone EfficientRep, βελτιώσεις στη δομή συνδυασμού χαρακτηριστικών (neck) (Rep-PAN) και κεφαλής (Efficient Decoupled Head)
YOLOv8 [20]	backbone CSPDarknet53, μονάδα C2f αντί για FPN (συνδυασμός χαρακτηριστικών διαφόρων επιπέδων)
YOLOv9 [41]	Generalized Efficient Layer Aggregation Network (GELAN), Programmable Gradient Information
YOLOv10 [39]	Διπλή ανάθεση κατηγοριών για την αποφυγή επεξεργασίας NMS (Non-Maximum Suppression)

2.1.2 Μοντέλα με βάση δίκτυα Transformer

Ο transformer [37] είναι η πρώτη αρχιτεκτονική μοντέλου ακολουθίας προς ακολουθία που βασίζεται αποκλειστικά στην προσοχή (attention) για την κατανόηση των εξαρτήσεων και την παραγωγή αποτελεσμάτων, χωρίς τη συνεισφορά βοηθητικών συστημάτων.

Το μοντέλο έχει μια δομή κωδικοποιητή-αποκωδικοποιητή, όπως φαίνεται στην παρακάτω εικόνα:

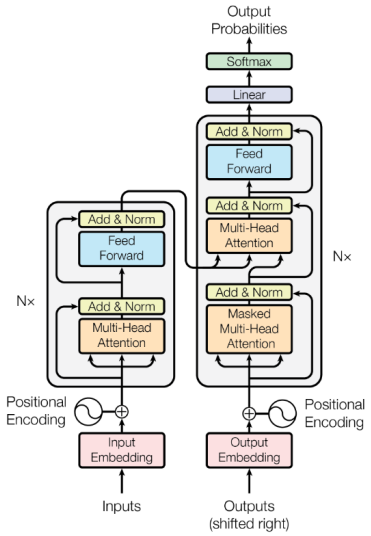


Figure 2.6: Η αρχιτεκτονική του Transformer (Εικόνα από [37])

Κάθε μπλοκ κωδικοποιητή αποτελείται από μία μονάδα προσοχής πολλαπλών κεφαλών (multi-head attention) που ακολουθείται από ένα πλήρως συνδεδεμένο δίκτυο (feed forward network). Η έξοδος κάθε από τα 2 αυτά υπο-modules συνδυάζεται με την είσοδό του χρησιμοποιώντας μια υπολειμματική σύνδεση (residual connection) [13] και κανονικοποίηση επιπέδου (layer normalization) [1] εφαρμόζεται στο αποτέλεσμα. Στην αρχική υλοποίηση, ο κωδικοποιητής αποτελείται από N=6 πανομοιότυπα μπλοκ.

Ένα μπλοκ αποκωδικοποιητή έχει παρόμοια δομή, με τη διαφορά ότι περιέχει ένα επιπλέον module προσοχής πολλαπλών κεφαλών ανάμεσα στην πρώτη μονάδα προσοχής και το πλήρως συνδεδεμένο δίκτυο, το οποίο δέχεται την έξοδο του κωδικοποιητή ως είσοδο. Το πρώτο επίπεδο προσοχής στην περίπτωση του αποκωδικοποιητή είναι "μασκαρισμένο"(masked), ώστε να μην μπορεί να λάβει υπόψη τις μελλοντικές εξόδους.

Τα modules προσοχής δέχονται ένα διάνυσμα ερωτήματος (query vector) και ένα σύνολο ζευγαριών διανυσμάτων κλειδιών-τιμών (key-value vectors) (που προέρχονται από embeddings που κωδικοποιούν τη "σημασία" τμημάτων της εισόδου) και παράγουν μια έξοδο, η οποία είναι ένα σταθμισμένο άθροισμα των τιμών, με τα βάρη να ορίζονται σύμφωνα με κάποια συνάρτηση συμβατότητας μεταξύ του ερωτήματος και του κλειδιού που αντιστοιχεί σε κάθε τιμή. Για να συμπεριληφθεί πληροφορία σχετικά με τη θέση κάθε τμήματος στην είσοδο, προστίθενται μοναδικά διανύσματα κωδικοποίησης θέσης (positional encoding vectors) στα embeddings. Αυτά τα διανύσματα έχουν το ίδιο μήκος με τα embeddings, και η τιμή της i-th θέσης του διανύσματος κωδικοποίησης θέσης για ένα τμήμα στη θέση p της εισόδου είναι $sin(\frac{p}{10000^{\frac{2i}{d_{emb}}}})$ ή $cos(\frac{p}{10000^{\frac{2i}{d_{emb}}}})$ για άρτιες και περιττές θέσεις i αντίστοιχα. Με αυτόν τον τρόπο, οι χαμηλότερες συχνότητες προσφέρουν διαφοροποίηση μεταξύ πιο απομακρυσμένων θέσεων, και οι υψηλότερες συχνότητες κάνουν το ίδιο για τμήματα της εισόδου που είναι πιο κοντά.

Αν Q είναι ενας πίνακας ερωτημάτων και K και V είναι οι αντίστοιχοι πίνακες κλειδιών και τιμών, τα αποτελέσματα της προσοχής μπορούν να υπολογιστούν ως:

$$A(Q, K, V) = softmax(\frac{Q \cdot K^T}{\sqrt{d_k}}) \cdot V \tag{2.1}$$

d_k είναι η διάσταση των διανυσμάτων ερωτήματος/κλειδιού (query/key vector), και η διαίρεση με $\sqrt{d_k}$ χρησιμοποιείται για να αποφευχθούν εξαιρετικά μεγάλες τιμές του παραπάνω εσωτερικού γινομένου.

Για καλύτερες επιδόσεις, οι παραπάνω λειτουργίες δεν εκτελούνται μόνο μία φορά, αλλά πολλές (προσοχή πολλαπλών κεφαλών - multi-head attention), με διαφορετικά βάρη για τα ερωτήματα, τα κλειδιά και τις τιμές, και η έξοδος υπολογίζεται ως μια προβολή της σύνθεσης των επιμέρους αποτελεσμάτων των κεφαλών προσοχής.

DETR

Ο DETR [5] είναι ένα μοντέλο σχεδιασμένο για ανίχνευση αντικειμένων, βασισμένο στην αρχιτεκτονική του transformer. Αποτελείται από ένα CNN backbone που εξάγει χαρακτηριστικά από την εικόνα εισόδου, έναν κωδικοποιητή transformer, έναν αποκωδικοποιητή transformer και ένα πλήρως συνδεδεμένο δίκτυο που παράγει τις τελικές προβλέψεις.

Ο χάρτης χαρακτηριστικών που περιέχει d κανάλια με ύψος H και πλάτος W, που παράγεται από το CNN, μετατρέπεται σε μια ακολουθία HxW διανυσμάτων μήκους d, και μετά την προσθήκη χωρικής κωδικοποίησης αυτή η ακολουθία περνάει ως είσοδος στον κωδικοποιητή. Ο αποκωδικοποιητής λαμβάνει N (εκπαιδευμένες) ερωτήσεις (queries) αντικειμένων ως είσοδο, "προσέχει" στην έξοδο του κωδικοποιητή, όπως στο αρχικό μοντέλο transformer, και παράγει N διανύσματα που περνούν από ένα κοινό πλήρως συνδεδεμένο δίκτυο για να προβλέψουν τα τελικά bounding boxes.

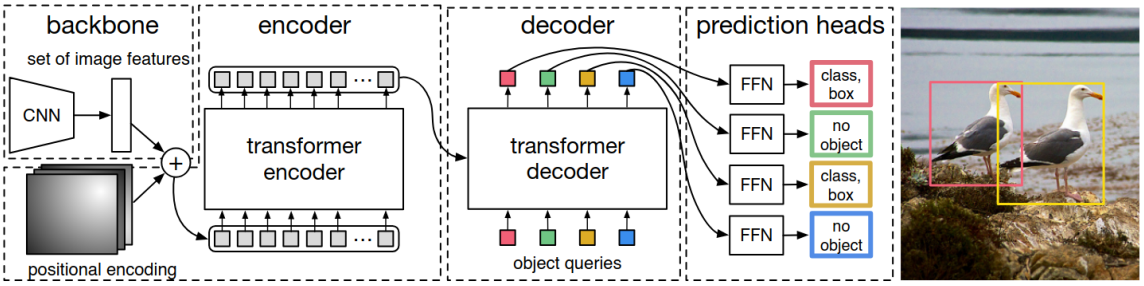


Figure 2.7: Μια επισκόπηση της αρχιτεκτονικής του DETR (εικόνα από [5])

Deformable DETR

Ο Deformable DETR [46] είναι ένα ακόμα μοντέλο ανίχνευσης αντικειμένων που βασίζεται στην αρχιτεκτονική του transformer, το οποίο προσπάθησε να αντιμετωπίσει δύο κύριες αδυναμίες του DETR. Συγκεκριμένα, πρώτα απ' όλα ο DETR πρέπει να προσέχει σε ολόκληρο τον χάρτη χαρακτηριστικών της εικόνας εισόδου για να προβλέψει τα bounding boxes. Αυτό κάνει το υπολογιστικό κόστος απαγορευτικό για πιο λεπτομερείς χάρτες χαρακτηριστικών, που θα επέτρεπαν στο μοντέλο να αποδίδει καλύτερα, ειδικά για μικρότερα αντικείμενα. Επιπλέον, η διαδικασία εκπαίδευσης του DETR χρειάζεται πολύ χρόνο για να ολοκληρωθεί, κυρίως λόγω της δυσκολίας εκπαίδευσης των βαρών προσοχής.

Για να αποφύγει αυτά τα ζητήματα, ο Deformable DETR εισήγαγε την μονάδα παραμορφώσιμης προσοχής (Deformable Attention module), με την αρχιτεκτονική που μπορεί να φανεί στο σχήμα παρακάτω:

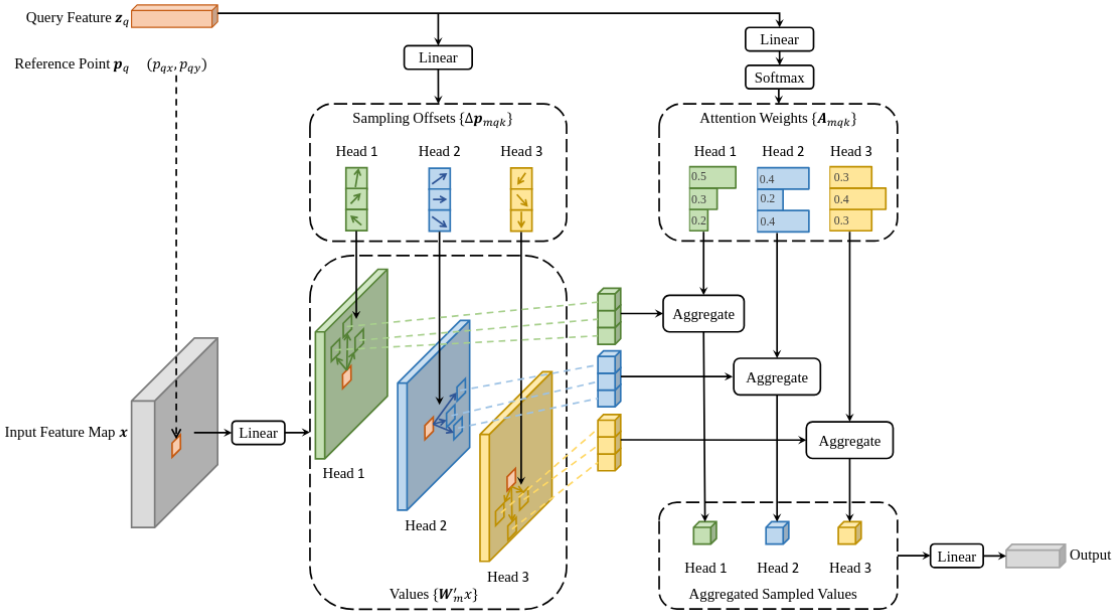


Figure 2.8: Η μονάδα παραμορφώσιμης προσοχής (εικόνα από [46])

Η έξοδος της μονάδας παραμορφώσιμης προσοχής μπορεί να εκφραστεί ως:

$$A(z_q, p_q, x) = \sum_{m=1}^{M} W_m \cdot [\sum_{k=1}^{K} A_{mqk} \cdot W'_m x(p_q + \Delta p_{mqk})] \tag{2.2}$$

όπου m είναι ο δείκτης κεφαλής προσοχής, k είναι ο δείκτης επιλεγμένου στοιχείου κλειδιού, και q είναι ο δείκτης στοιχείου ερωτήματος, με z_q και p_q να αναπαριστούν το διάνυσμα χαρακτηριστικών και τη θέση του στοιχείου ερωτήματος, αντίστοιχα, στον χάρτη χαρακτηριστικών. Δp_{mqk} είναι η απόσταση από το αρχικό στοιχείο ερωτήματος για την κεφαλή m, υποδεικνύοντας τη θέση του k-στου κλειδιού που θα προσέξει το μοντέλο.

Το τελικό αποτέλεσμα είναι ένα σταθμισμένο άθροισμα των αποτελεσμάτων που παράγονται από τις κεφαλές προσοχής, όπου το αποτέλεσμα κάθε κεφαλής είναι ένα σταθμισμένο άθροισμα των τιμών που αντιστοιχούν στις επιλεγμένες τοποθεσίες. Τα βάρη για αυτό το άθροισμα είναι οι τιμές προσοχής A_{mqk}. Τόσο οι A_{mqk} όσο και οι Δp_{mqk} υπολογίζονται από το διάνυσμα χαρακτηριστικών z_q του ερωτήματος χρησιμοποιώντας ένα εκπαιδευμένο επίπεδο γραμμικής προβολής.

Αυτό μπορεί να επεκταθεί για πολλαπλές κλίμακες, δίνοντας το παρακάτω αποτέλεσμα:

$$A(z_q, p_q, x^l) = \sum_{m=1}^{M} W_m \cdot [\sum_{l=1}^{L} \sum_{k=1}^{K} A_{mqkl} \cdot W'_m x^l(\phi_l(p_q) + \Delta p_{mqkl})] \tag{2.3}$$

όπου l είναι ο δείκτης κλίμακας και το $\phi_l(p_q)$ είναι μια αναπροσαρμογή κλίμακας (rescaling) των συντεταγμένων p_q για τον χάρτη χαρακτηριστικών της κλίμακας l.

Για τον κωδικοποιητή του μοντέλου, οι κανονικές μονάδες προσοχής αντικαθίστανται με τις παραμορφώσιμες, οι οποίες επεξεργάζονται πολλαπλά επίπεδα κλιμάκων χαρακτηριστικών (4 στην αρχική υλοποίηση) που εξάγονται από την εικόνα εισόδου. Στην περίπτωση του αποκωδικοποιητή, τα επίπεδα αυτοπροσοχής (self-attention) δεν τροποποιούνται, καθώς όλες οι είσοδοί του είναι ερωτήματα αντικειμένων (object queries). Ωστόσο, τα επίπεδα διασταυρούμενης προσοχής (cross-attention) προσέχουν στους χάρτες χαρακτηριστικών που παράγονται από τον κωδικοποιητή. Συνεπώς, η μονάδα διασταυρούμενης προσοχής μπορεί να αντικατασταθεί από μια παραμορφώσιμη. Το αντίστοιχο της εισόδου ερωτήματος της μονάδας είναι ένα ερώτημα αντικειμένου, οπότε το σημείο αναφοράς p_q θα πρέπει να προκύψει από αυτά τα ερωτήματα με έναν εκπαιδευμένο γραμμικό προβολέα.

RT-DETR

Ο RT-DETR [30] είναι το πρώτο μοντέλο βασισμένο στους transformers που επιτυγχάνει επιδόσεις πραγματικού χρόνου, μαζί με πολύ υψηλά σκορ ακρίβειας, παρόμοια με αυτά των τελευταίων μοντέλων YOLO. Η βελτιωμένη απόδοσή του σε σύγκριση με τους προηγούμενους ανιχνευτές βασισμένους σε transformer είναι το αποτέλεσμα του επανασχεδιασμού του υποσυστήματος του κωδικοποιητή, το οποίο ήταν το υπολογιστικό σημείο συμφόρησης των παλαιότερων αρχιτεκτονικών.

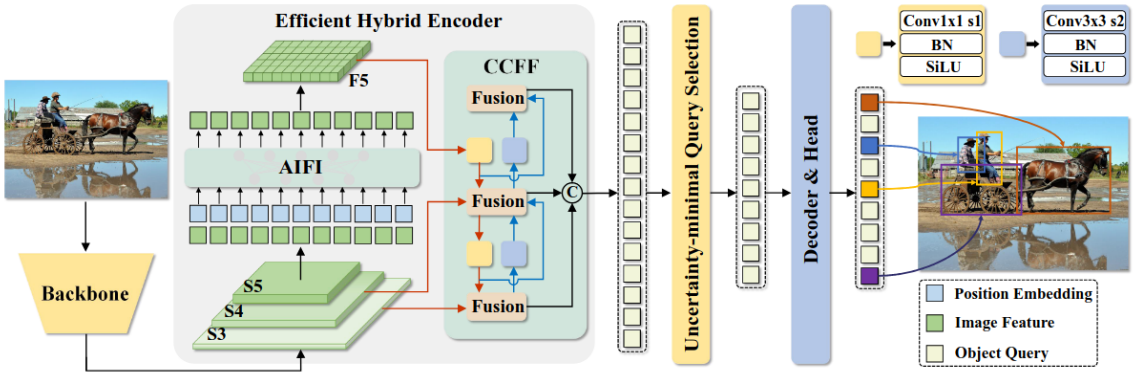


Figure 2.9: Η δομή του RT-DETR (εικόνα από [30])

Ο επανασχεδιασμένος κωδικοποιητής (αποδοτικός υβριδικός κωδικοποιητής) αποτελείται από 2 νέα υποσυστήματα, AIFI (Attention-based Intra-scale Feature Interactio) και CCFF (CNN-based Cross-scale Feature Fusion). Ο κωδικοποιητής δέχεται χάρτες χαρακτηριστικών 3 κλιμάκων, και το υποσύστημα AIFI εφαρμόζει αυτοπροσοχή μόνο στο τελευταίο (χαμηλότερης ανάλυσης) επίπεδο, καθώς η αλληλεπίδραση μεταξύ χαρακτηριστικών χαμηλότερων επιπέδων δεν συμβάλλει σημαντικά στην κατανόηση των σημασιολογικών σχέσεων μεταξύ των τμημάτων της εικόνας. Η λειτουργία του CCFF βασίζεται στο μπλοκ συγχώνευσης, το οποίο συγχωνεύει τα χαρακτηριστικά 2 κλιμάκων σε ένα νέο χαρακτηριστικό και έχει τη δομή που μπορεί να φανεί

στο παρακάτω σχήμα. Από την έξοδο του κωδικοποιητή, το μοντέλο επιλέγει έναν σταθερό αριθμό κατάλληλων χαρακτηριστικών ως ερωτήματα αντικειμένων για τον κωδικοποιητή, προκειμένου να αποφύγει τη δυσκολία που παρουσιαζόταν κατά την προσπάθεια βελτιστοποίησης των ερωτημάτων στα παλαιότερα μοντέλα.

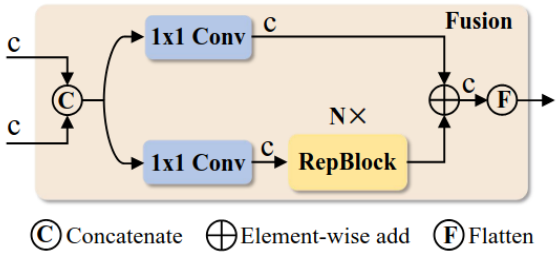


Figure 2.10: Το μπλοκ συγχώνευσης του υποσυστήματος CCFF (εικόνα από [30])

2.2 Παρακολούθηση (Tracking) Αντικειμένων

Η παρακολούθηση αντικειμένων είναι ένα ακόμα σημαντικό αντικείμενο της όρασης υπολογιστών, και περιλαμβάνει την απόδοση ταυτοτήτων στα ανιχνευμένα αντικείμενα σε ένα καρέ βίντεο, και τον εντοπισμό των νέων τους θέσεων στα επόμενα καρέ, διατηρώντας τις μοναδικές τους ταυτότητες. Η παρακολούθηση μπορεί να πραγματοποιηθεί χρησιμοποιώντας παραδοσιακούς αλγορίθμους όρασης υπολογιστών, η λειτουργία των οποίων βασίζεται κυρίως στον εντοπισμό αλλαγών μεταξύ των καρέ, ή αλγορίθμους βασισμένους στη βαθιά μάθηση, που μπορούν, για παράδειγμα, να λάβουν υπόψη χαρακτηριστικά των αντικειμένων ενδιαφέροντος.

2.2.1 "Παραδοσιακοί" Trackers

KCF Tracker

Ο Kernelized Correlation Filter (KCF) Tracker [16] προσπαθεί να μειώσει τις απαιτήσεις χρόνου και μνήμης της διαδικασίας παρακολούθησης, αποφεύγοντας εκπτώσεις στην ποιότητα των αποτελεσμάτων. Αυτό επιτυγχάνεται με τη χρήση των ιδιοτήτων των κυκλοτερών μητρώων (circulant matrices). Θετικά και αρνητικά δείγματα (τμήματα της εικόνας) μοντελοποιούνται ως τέτοια μητρώα με συνεχόμενες μετατοπίσεις του θετικού δείγματος. Το προκύπτον (κυκλοτερές) μητρώο για ένα απλό δισδιάστατο διάνυσμα θα έχει την ακόλουθη μορφή:

$$X = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ x_n & x_1 & \dots & x_{n-1} \\ x_{n-1} & x_n & \dots & x_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ x_2 & x_3 & \dots & x_1 \end{bmatrix}$$

Χρησιμοποιώντας τα παραπάνω δεδομένα, μπορεί να επιτευχθεί καλύτερη απόδοση χαρτογραφώντας τις εισόδους σε έναν χώρο χαρακτηριστικών υψηλότερης διάστασης και μοντελοποιώντας την ομοιότητά τους ως το εσωτερικό γινόμενο των προβολών τους, χρησιμοποιώντας το κόλπο του πυρήνα [35]. Αυτά τα εσωτερικά γινόμενα αποθηκεύονται σε έναν πίνακα nxn, που ονομάζεται μητρώο πυρήνα. Η αναφερόμενη μορφή δεδομένων οδηγεί σε μητρώα πυρήνα που είναι επίσης κυκλοτερή για πολλές χρήσιμες συναρτήσεις πυρήνα. Αυτό καθιστά δυνατή την αξιοποίηση της ιδιότητας αυτών των μητρώων να γίνονται διαγώνια όταν εφαρμόζεται Διακριτός Μετασχηματισμός Fourier, οδηγώντας σε ταχύτερους, στοιχείο προς στοιχείο υπολογισμούς στον χώρο Fourier, στον οποίο οι συγγραφείς εργάστηκαν για να αποκτήσουν την τελική λύση για τις παραμέτρους του tracker όπως περιγράφεται στο αναφερόμενο άρθρο.

CSRT Tracker

Ο CSRT tracker που παρέχεται από την OpenCV είναι μια υλοποίηση της μεθόδου CSR-DCF που προτάθηκε στο [29]. Η θέση του παρακολουθούμενου αντικειμένου εκτιμάται με βάση τη μεγιστοποίηση της πιθανότητας $p(x|h) = \sum_{d=1}^{N_d} p(x|f_d)p(f_d)$, όπου N_d είναι ο αριθμός των καναλιών, το d είναι ο δείκτης καναλιού, f_d είναι ένας χάρτης χαρακτηριστικών που αντιστοιχεί στη διάσταση d, το $p(f_d)$ αντιπροσωπεύει την αξιοπιστία του καναλιού και $p(x|f_d) = [f_d * h_d](x)$, όπου h_d είναι ένα εκπαιδευμένο φίλτρο. Αυτά τα φίλτρα εκτιμώνται ως $\arg\min_h \sum_{d=1}^{N_d} ||f_d * h_d - g||^2 + \lambda \sum_{d=1}^{N_d} ||h_d||^2$, όπου το g είναι το επιθυμητό αποτέλεσμα. Αυτή η εκτίμηση υποστηρίζεται από χωρικούς χάρτες αξιοπιστίας, οι οποίοι υποδεικνύουν ποια εικονοστοιχεία είναι κατάλληλα για να συμβάλλουν στη διαδικασία εκπαίδευσης, με βάση το αν γίνονται αντιληπτά ως προσκήνιο ή υπόβαθρο.

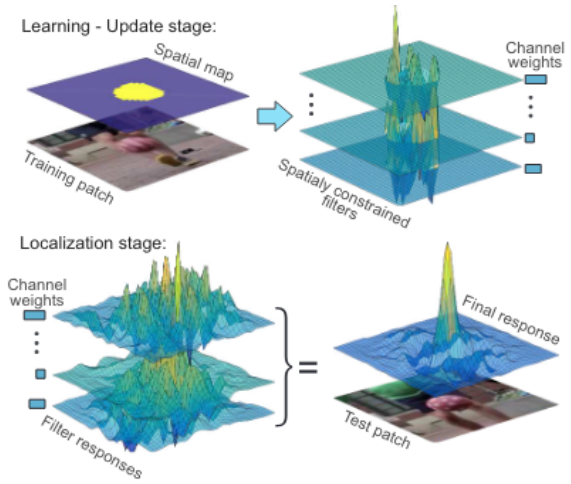


Figure 2.11: Επισκόπηση της μεθόδου CSR-DCF (Εικόνα από [29])

Για έναν κύκλο της διαδικασίας παρακολούθησης, το αντικείμενο εντοπίζεται με βάση τις

εξόδους των φίλτρων, σταθμισμένες με τις βαθμολογίες αξιοπιστίας ανά κανάλι, οι οποίες λαμβάνονται σύμφωνα με τις τιμές που παράγονται από τα φίλτρα που αντιστοιχούν σε κάθε κανάλι, και τα φίλτρα ενημερώνονται με βάση τον νέο εντοπισμό.

MOSSE Tracker

Ο Minimum Output Sum of Squared Error (MOSSE) [4] είναι ένας άλλος tracker που λειτουργεί χρησιμοποιώντας συσχετιστικά φίλτρα. Κατά τη διάρκεια της φάσης αρχικοποίησης/εκπαίδευσης, χρησιμοποιώντας ένα σύνολο i εκπαιδευτικών εικόνων, εκτιμά ένα φίλτρο h που μπορεί να ελαχιστοποιήσει το $\sum_i |F_i \odot H^* - G_i|^2$, όπου F_i είναι ο μετασχηματισμός Fourier της i-οστής μήτρας χαρακτηριστικών, G_i είναι ο μετασχηματισμός Fourier της αναμενόμενης εξόδου g_i και H^* είναι ο μιγαδικός συζυγής του μετασχηματισμού Fourier του φίλτρου h. Το $\odot$ αντιπροσωπεύει τον στοιχείο προς στοιχείο πολλαπλασιασμό. Οι περισσότεροι από τους υπολογισμούς πραγματοποιούνται στον χώρο Fourier, καθώς αυτό επιτρέπει τη διεξαγωγή των συνελίξεων ως στοιχείο προς στοιχείο πολλαπλασιασμούς. Το παραπάνω πρόβλημα ελαχιστοποίησης οδηγεί στο αποτέλεσμα:

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^*} \tag{2.4}$$

Κατά τη διάρκεια της φάσης ενημέρωσης, το φίλτρο ενημερώνεται ως εξής, διατηρώντας κάποιες πληροφορίες από τα παλαιότερα καρέ, αλλά εστιάζοντας κυρίως στα πρόσφατα:

$$H_i^* = \frac{A_i}{B_i} \tag{2.5}$$

$$A_i = \eta G_i \odot F_i^* + (1-\eta) A_{i-1} \tag{2.6}$$

$$B_i = \eta F_i \odot F_i^* + (1-\eta) B_{i-1} \tag{2.7}$$

Η τιμή $\eta = 0.125$ βρέθηκε να δίνει τα καλύτερα αποτελέσματα.

MedianFlow Tracker

Ο MedianFlow tracker [21] βασίζεται σε μια διαδικασία εκτίμησης σφάλματος που ονομάστηκε forward-backward error. Υποθέτοντας ότι ένας tracker εκτιμά την πορεία ενός σημείου ως $T_f = (x_t, x_{t+1}, .., x_{t+k})$ για μια ακολουθία k καρέ, μπορεί να παραχθεί μια οπισθοδρομική πορεία για την αντίστροφη ακολουθία καρέ, $T_b = (\hat{x_t}, \hat{x_{t+1}}, .., \hat{x_{t+k}})$. Το forward-backward error ορίζεται τότε ως η απόσταση μεταξύ των T_f και T_b (η ευκλείδεια απόσταση χρησιμοποιήθηκε στην αρχική εργασία ως απλό παράδειγμα).

Ο MedianFlow tracker δέχεται 2 εικόνες I_t και I_{t+1} ως είσοδο, με ένα bounding box b_t που αποδίδεται στην πρώτη, υποδεικνύοντας το αντικείμενο που θα παρακολουθηθεί. Ένα σύνολο σημείων προσδιορίζεται μέσα σε αυτό το κουτί, τα οποία στη συνέχεια παρακολουθούνται χρησιμοποιώντας τη μέθοδο οπτικής ροής Lucas-Kanade [28]. Το σφάλμα των εκτιμημένων

διαδρομών υπολογίζεται όπως περιγράφηκε παραπάνω, το 50% των σημείων (τα οποία έχουν το υψηλότερο σφάλμα) απορρίπτονται, και ένα νέο bounding box b_{t+1} εκτιμάται για το I_{t+1}.

2.2.2 Trackers Βαθιάς Μάθησης

GOTURN Tracker

Ο GOTURN tracker [15] χρησιμοποιεί τη σύγκριση βαθιών χαρακτηριστικών των εικόνων εισόδου προκειμένου να εντοπίσει τον καθορισμένο στόχο σε διαδοχικά καρέ. Οι είσοδοι του tracker είναι 2 κομμένα τμήματα διαδοχικών καρέ, με το πρώτο να περιέχει το αντικείμενο προς παρακολούθηση μαζί με κάποια καθορισμένη περιοχή συμφραζομένων και το δεύτερο είναι μια περιοχή της εικόνας στην οποία εκτιμάται ότι βρίσκεται ο στόχος.

Αυτά τα τμήματα περνούν στη συνέχεια μέσα από μια ακολουθία συνελικτικών στρωμάτων, η έξοδος των οποίων είναι με τη σειρά της η είσοδος μιας ακολουθίας πλήρως συνδεδεμένων στρωμάτων, που προσπαθούν να εκτιμήσουν τη νέα θέση του παρακολουθούμενου αντικειμένου. Στην αρχική υλοποίηση, τα συνελικτικά στρώματα είναι τα προεκπαιδευμένα πρώτα 5 στρώματα του δικτύου CaffeNet [18], ενώ τα πλήρως συνδεδεμένα στρώματα (3 στρώματα με 4096 νευρώνες το καθένα) εκπαιδεύονται σε ένα σύνολο βίντεο και στατικών εικόνων. Στη δεύτερη περίπτωση, η κίνηση προσομοιώνεται τροφοδοτώντας τυχαία κομμένα τμήματα από την εικόνα στο δίκτυο.

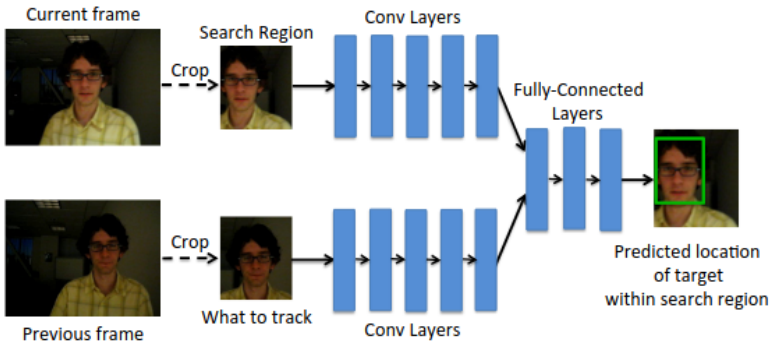


Figure 2.12: Η αρχιτεκτονική του GOTURN (Εικόνα από [15])

DeepSort Tracker

Ο DeepSORT [43] είναι ένας αλγόριθμος παρακολούθησης πολλών αντικειμένων μέσω ανίχνευσης που βασίζεται στον SORT [2], χρησιμοποιώντας βαθιά χαρακτηριστικά των εικόνων για να βελτιώσει την απόδοσή του. Η κατάσταση των τροχιών (tracks) μπορεί να αναπαρασταθεί χρησιμοποιώντας 8 μεταβλητές, u,v,γ,h,$\dot{u}$,$\dot{v}$,$\dot{\gamma}$,$\dot{h}$, όπου (u,v) είναι το κέντρο του bounding box σε μορφή συντεταγμένων (x,y), γ είναι η αναλογία διαστάσεών του και h είναι το ύψος του, με τις υπόλοιπες 4 μεταβλητές να είναι ο ρυθμός μεταβολής τους.

Με μια παρατήρηση των u,v,γ και h, γίνεται μια αρχική εκτίμηση των νέων καταστάσεων αντικειμένων χρησιμοποιώντας ένα φίλτρο Kalman. Νέες ανιχνεύσεις ταιριάζονται στη συνέχεια

με τις εκτιμήσεις των τροχιών χρησιμοποιώντας τον Hungarian Algorithm [22], με 2 μετρικές απόστασης που πρέπει να ελαχιστοποιηθούν: Την τετραγωνική απόσταση Mahalanobis μεταξύ των εκτιμήσεων του φίλτρου Kalman και των νέων ανιχνεύσεων, και μια νέα "απόσταση" εμφάνισης. Αυτή η απόσταση εμφάνισης είναι ίση με 1 - την ομοιότητα συνημίτονου μεταξύ των διανυσμάτων χαρακτηριστικών των ανιχνεύσεων και των προβλέψεων του φίλτρου Kalman. Τα διανύσματα χαρακτηριστικών εξάγονται από τα τμήματα της εικόνας χρησιμοποιώντας ένα προεκπαιδευμένο συνελικτικό δίκτυο.

2.3 Πρόβλεψη Χρονοσειρών

2.3.1 Δίκτυα LSTM

Τα δίκτυα LSTM (Long-Short Term Memory) [17] είναι Αναδρομικά Νευρωνικά Δίκτυα (RNNs) που έχουν σχεδιαστεί για να λαμβάνουν υπόψη τόσο τις μακροχρόνιες όσο και τις βραχυχρόνιες εξαρτήσεις μεταξύ των στοιχείων των ακολουθιών εισόδου. Τα RNNs είναι νευρωνικά δίκτυα που επεξεργάζονται στοιχεία μιας ακολουθίας χρησιμοποιώντας την έξοδο για το στοιχείο e_{t-1} ως είσοδο για να υπολογίσουν την έξοδο για το στοιχείο e_t, έχοντας έτσι τη συνατότητα να "θυμούνται" πληροφορίες που συνάντησαν νωρίτερα κατά τη διάρκεια της επεξεργασίας αυτής της ακολουθίας. Εάν αυτός ο βρόχος "ξεδιπλωθεί", ένα RNN μπορεί να απεικονιστεί με την ακόλουθη μορφή:

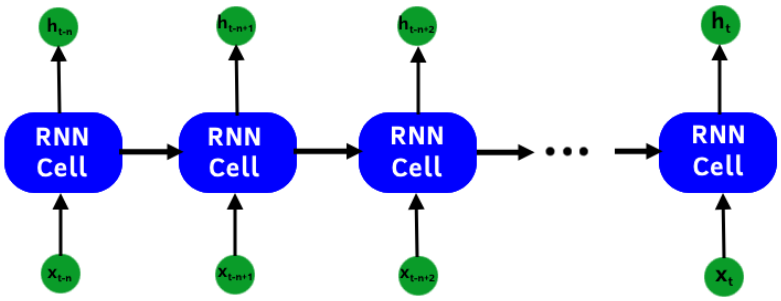


Figure 2.13: Αναπαράσταση ενός RNN

Στην περίπτωση των LSTMs, η δομή των κελιών μπορεί να θεωρηθεί ότι αποτελείται από 3 διακριτά μέρη (πύλες): την πύλη λήθης (forget gate), την πύλη εισόδου (input gate) και την πύλη εξόδου (output gate). Μια επισκόπηση της εσωτερικής δομής ενός κελιού LSTM μπορεί να παρατηρηθεί στην παρακάτω εικόνα:

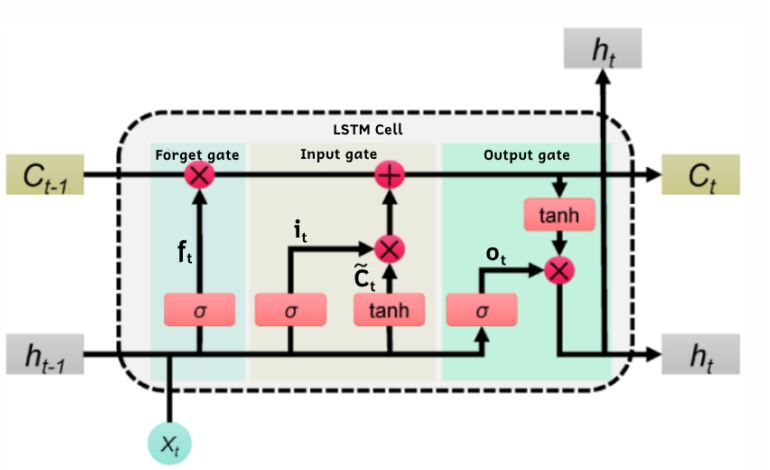


Figure 2.14: Αναπαράσταση της δομής ενός κελιού LSTM (εικόνα από mlarchive.com)

Το μέρος αυτής της αρχιτεκτονικής που επιτρέπει στο δίκτυο να έχει πρόσβαση σε παλαιότερες πληροφορίες είναι η κατάσταση του κελιού (C_t στό σχήμα παραπάνω), η οποία δέχεται μικρής κλίμακας αλλαγές για κάθε νέα είσοδο. Η πρώτη ενημέρωση της κατάστασης του κελιού πραγματοποιείται από την πύλη λήθης, η οποία ρυθμίζει το ποσοστό κάθε στοιχείου της τελευταίας κατάστασης του κελιού που θα διατηρηθεί. Αυτό επιτυγχάνεται με ένα στοιχείο προς στοιχείο γινόμενο της παλιάς κατάστασης με έναν πίνακα λήθης, που υπολογίζεται ως εξής:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.8}$$

όπου W_f είναι ένας εκπαιδευμένος πίνακας βαρών, h_{t-1} είναι η προηγούμενη κρυφή κατάσταση, b_f είναι ενα βάρος πόλωσης (bias) και σ είναι η συνάρτηση sigmoid.

Η πύλη εισόδου στη συνέχεια ενημερώνει την κατάσταση του κελιού με νέες πληροφορίες, προσθέτοντας επιλεκτικά μέρος της εισόδου και της τελευταίας κρυφής κατάστασης:

$$\widetilde{C_t} = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.9}$$

$$i_t = tanh(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.10}$$

Οι νέες πληροφορίες που προστίθενται στην κατάσταση του κελιού είναι το στοιχείο προς στοιχείο γινόμενο των $\widetilde{C_t}$ και $\widetilde{i_t}$. Η νέα κατάσταση του κελιού μετά από τις δύο αυτές ενημερώσεις είναι:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t} \tag{2.11}$$

Τέλος, η έξοδος που παράγεται από την πύλη εξόδου (η οποία είναι επίσης η επόμενη κρυφή κατάσταση) συνδυάζει τη νέα κατάσταση του κελιού και την τελευταία κρυφή κατάσταση ως εξής:

$$h_t = o_t \odot tanh(C_t) \tag{2.12}$$

όπου

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.13}$$

Ενώ τα LSTMs έχουν κάποια ικανότητα να αναγνωρίζουν μακροχρόνιες εξαρτήσεις, δεν μπορούν να το κάνουν όσο αποτελεσματικά τα μοντέλα transformer. Είναι επίσης πιο επιρρεπή στην υπερπροσαρμογή (overfitting) όταν εκπαιδεύονται υπερβολικά σε ένα μικρό σύνολο δεδομένων. Ωστόσο, είναι αξιόπιστα για εργασίες με χαμηλότερες απαιτήσεις ακρίβειας και σχετικά χαμηλή πολυπλοκότητα, κάτι που τα καθιστά μια αξιόλογη εναλλακτική σε πολλές περιπτώσεις.

Κεφάλαιο 3

Προτεινόμενη Αρχιτεκτονική

Η προτεινόμενη λύση για το πρόβλημα που περιγράφτηκε βασίζεται σε μια δομή επεξεργασίας 2 τμημάτων. Ένα πρώτο, ελαφρύτερο μέρος των απαραίτητων υπολογισμών πραγματοποιείται απευθείας στο UAV, ενώ το δεύτερο, βαρύτερο και πιο ακριβές ανατίθεται σε έναν ισχυρό απομακρυσμένο υπολογιστή, ο οποίος θα επικοινωνεί συνεχώς με το UAV και θα παρέχει υποστηρικτικές οδηγίες.

3.1 UAV

3.1.1 Υλικό

Τα υλικά μέρη του τετρακόπτερου UAV που χρησιμοποιήθηκαν στην υλοποίηση του συστήματος είναι τα εξής:

Αυτόματος Πιλότος

Για την πλοήγηση του UAV, χρησιμοποιήθηκε ο αυτόματος πιλότος Cube Orange. Είναι εξοπλισμένος με έναν ισχυρό επεξεργαστή (32bit ARM STM32H753 Cortex-M7) ικανό να ελέγχει την πτήση του τετρακόπτερου με βάση τις εισόδους διαφόρων αισθητήρων που είναι συνδεδεμένοι σε αυτόν και τις εντολές πλοήγησης που στέλνει ο χρήστης μέσω ενός συστήματος τηλεχειρισμού. Η επικοινωνία με το σύστημα αυτό πραγματοποιείται μέσω σειριακής θύρας, χρησιμοποιώντας το πρωτόκολλο μηνυμάτων Mavlink 2. Ο αυτόματος πιλότος μπορεί επίσης να δέχεται εντολές Mavlink μέσω μιας δευτερεύουσας θύρας, κάτι που θα είναι χρήσιμο για την αυτόματη καθοδήγηση του UAV σε περίπτωση που εντοπιστεί ένας στόχος. Το λογισμικό ardupilot που χρησιμοποιεί ο πιλότος διαθέτει μεγάλο αριθμό παραμέτρων, που μπορούν να ρυθμιστούν κατάλληλα. Μπορεί επίσης να λειτουργήσει σε διάφορα modes, τα οποία προσδιορίζουν τη συμπεριφορά του UAV σε διάφορες καταστάσεις και την ανταπόκρισή του του σε εισερχόμενες εντολές.

Κάμερα

Η είσοδος βίντεο του συστήματος ανίχνευσης/παρακολούθησης προέρχεται από μια κάμερα SIYI A8 Mini. Έχει τη δυνατότητα να παρέχει βίντεο υψηλής ανάλυσης (έως 4K) και υψηλή ευαισθησία στο φως, που της επιτρέπει να καταγράφει λεπτομερή καρέ ακόμα και σε πολύ χαμηλές συνθήκες φωτισμού.

Η κάμερα είναι τοποθετημένη σε ένα γυροσκοπικό σύστημα ικανό να περιστρέφεται γύρω από 3 άξονες, με εύρη pitch, yaw και roll -135^o- $+45^o$, -160^o- $+160^o$ και -30^o- $+30^o$ αντίστοιχα. Το γυροσκοπικό σύστημα λειτουργεί σε 3 modes, follow, lock και fpv, που του επιτρέπουν να ακολουθεί αργά την περιστροφή του UAV, να παραμένει στην αρχική του κατεύθυνση ή να κινείται ταυτόχρονα με αυτό. Σταθεροποιεί την είσοδο της κάμερας, αντιστεκόμενο σε περιστροφές του σώματος του UAV, κάτι που είναι εξαιρετικά σημαντικό στην παρούσα περίπτωση, καθώς ξαφνικές αλλαγές στη γωνία του βίντεο εισόδου θα καθιστούσαν αδύνατη τη διαδικασία tracking, ειδικά όταν το UAV προσαρμόζει την ταχύτητά του.

Η έξοδος βίντεο μπορεί να παρέχεται με 3 τρόπους, ethernet, HDMI ή CVBS. Ο προεπιλεγμένος τρόπος παροχής της εξόδου είναι μέσω ethernet ως ροή RTSP. Η κάμερα λειτουργεί ως διακομιστής (server) και καθιστά τη ροή αυτή διαθέσιμη σε μια προεπιλεγμένη διεύθυνση (rtsp://192.168.144.25:8554/main.264).

Τηλεχειρισμός

Για τον τηλεχειρισμό του UAV χρησιμοποιήθηκε το σύστημα Herelink. Ο σταθμός εδάφους του συστήματος ανταλλάσσει εντολές Mavlink με την onboard μονάδα στο UAV, ενώ υποστηρίζεται και η μετάδοση/λήψη δεδομένων βίντεο. Εκτός από τις εντολές πλοήγησης, το Herelink επιτρέπει τη μετάδοση εντολών για την αλλαγή της λειτουργίας πτήσης (mode) του αυτόματου πιλότου.

Επεξεργασία Δεδομένων

Η επεξεργασία των δεδομένων βίντεο στο UAV πραγματοποιείται σε ένα Raspberry Pi 4b (8GB RAM). Είναι ένας υπολογιστής μιας πλακέτας, ο οποίος λειτουργεί με έναν τετραπύρηνο επεξεργαστή 64-bit ARM-Cortex A72, που λειτουργεί στα 1.5GHz. Το Raspberry Pi χρησιμοποιεί μια κάρτα micro SD ως σκληρό δίσκο και λειτουργεί με το Raspberry Pi OS (προηγουμένως γνωστό ως Raspbian), το οποίο είναι μια διανομή Linux σχεδιασμένη για τους συγκεκριμένους υπολογιστές. Η συσκευή παρέχει μια θύρα ethernet, 2 USB3 και 2 USB2 θύρες, καθώς και ένα σύνολο 40 ακίδων, από τις οποίες 2 (RX/TX) μπορούν να χρησιμοποιηθούν για σειριακή επικοινωνία. Διαθέτει επίσης δυνατότητα σύνδεσης Wi-Fi, η οποία μπορεί να ενεργοποιηθεί αυτόματα για γνωστά δίκτυα.

Ένας βασικός παράγοντας για την επιτυχή υλοποίηση του συζητούμενου συστήματος είναι η ευαίσθητη και χρονικά αποδοτική διαδικασία ανίχνευσης αντικειμένων. Αυτό δεν μπορεί να πραγματοποιηθεί απευθείας στο Raspberry Pi, λόγω προφανών ελλείψεων υπολογιστικών πόρων. Επομένως, επιλέχθηκε το Intel Neural Compute Stick 2 για την συγκεκριμένη εργασία.

Η λειτουργία του NCS 2 βασίζεται σε έναν επιταχυντή υλικού που μπορεί να υπολογίζει την έξοδο βαθιών νευρωνικών δικτύων χωρίς τους περιορισμούς της CPU της πλακέτας στην

οποία είναι συνδεδεμένο. Αυτοί οι υπολογισμοί πραγματοποιούνται χρησιμοποιώντας το API της Intel OpenVINO inference engine, το οποίο δέχεται το μοντέλο σε μορφή OpenVINO IR (ένα αρχείο xml και ένα αρχείο bin, που κωδικοποιούν τη δομή και τα βάρη του μοντέλου). Δεδομένου ότι η κυκλοφορία του NCS 2 έχει διακοπεί, και οι τελευταίες εκδόσεις του OpenVINO δεν το υποστηρίζουν, χρησιμοποιήθηκε μια παλαιότερη έκδοση του εργαλείου (2021.4.2).

Τα καρέ από την έξοδο HDMI της κάμερας λαμβάνονται από το Raspberry Pi μέσω USB, χρησιμοποιώντας τον κατάλληλο προσαρμογέα (κάρτα βίντεο HDMI σε USB). Η λήψη της ροής βίντεο με τον προεπιλεγμένο τρόπο (RTSP μέσω ethernet) δεν ήταν δυνατή, καθώς για τη σωστή αποκωδικοποίησή της απαιτείται η λήψη όλων των καρέ-κλειδιών (key frames), γεγονός που προκαλεί σημαντική καθυστέρηση, λόγω της αδυναμίας του Raspberry Pi να αποκωδικοποιήσει τη ροή με επαρκή ταχύτητα, ακόμη και με ένα αποκλειστικό thread ανάγνωσης.

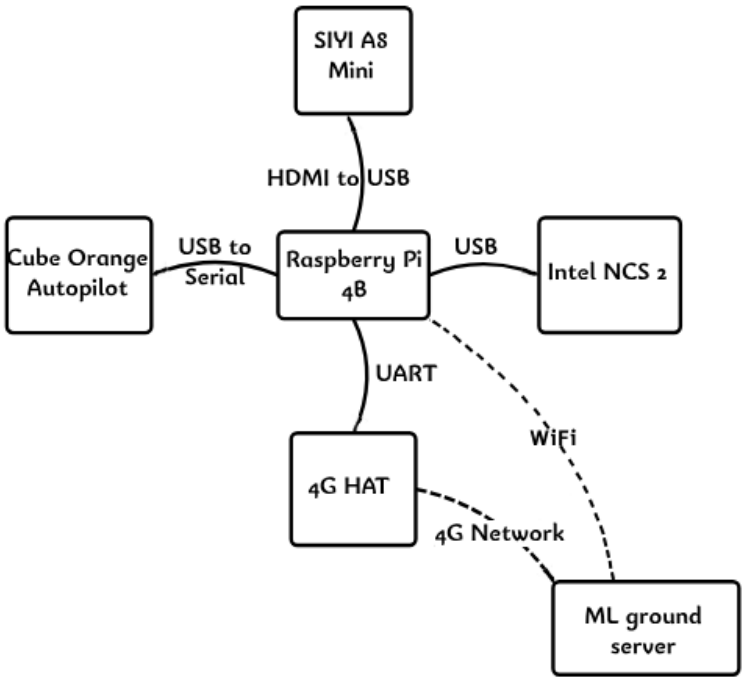


Figure 3.1: Σχηματική αναπαράσταση των τμημάτων του συστήματος και των μεταξύ τους συνδέσεων

Τα καταγεγραμμένα καρέ θα χρειαστεί επίσης να αποσταλούν στον απομακρυσμένο server. Σε κατάλληλες συνθήκες λειτουργίας, αυτό θα γίνει χρησιμοποιώντας το module Wi-Fi του Raspberry Pi, υποθέτοντας ότι το UAV θα πετάει σε έναν καθορισμένο χώρο με κάλυψη υψηλής ταχύτητας ασύρματου δικτύου. Ωστόσο, αν αυτό δεν είναι εφικτό, το Raspberry είναι εξοπλισμένο με ένα 4G HAT, με το οποίο επικοινωνεί μέσω UART. Φυσικά, η χρήση αυτής της εναλλακτικής λύσης δεν έχει την ίδια ταχύτητα μεταφοράς, με αποτέλεσμα χαμηλότερους ρυθμούς καρέ ανά δευτερόλεπτο για τον server. Το baud rate για την επικοινωνία μεταξύ του

Raspberry και του 4G HAT έχει οριστεί στην μέγιστη τιμή που επιτρέπεται από την πλακέτα (3Mbps), η οποία είναι επίσης κοντά στην μέγιστη τιμή που υποστηρίζεται από το HAT (περίπου 3.5Mbps).

Μετά από έναν κύκλο επεξεργασίας δεδομένων (όπως περιγράφεται παρακάτω), το Raspberry Pi θα πρέπει να καθοδηγήσει το UAV προς τον καθορισμένο στόχο. Αυτό επιτυγχάνεται με τη σύνδεση μεταξύ μιας θύρας USB του υπολογιστή και της δευτερεύουσας θύρας τηλεμετρίας του αυτόματου πιλότου, χρησιμοποιώντας έναν μετατροπέα USB σε σειριακή επικοινωνία, μέσω του οποίου το Raspberry θα μπορεί να στείλει τις απαραίτητες εντολές Mavlink.

3.1.2 Δομή του Λογισμικού

Ο σκοπός του λογισμικού που εκτελείται στο UAV είναι να ανιχνεύει οχήματα στο οπτικό του πεδίο, να επιλέγει ένα από αυτά ως στόχο και να το παρακολουθεί, ελαχιστοποιώντας τις καθυστερήσεις. Επιπλέον, πρέπει να είναι σε θέση να ενημερώνει την τοποθεσία του στόχου του σύμφωνα με τις οδηγίες του απομακρυσμένου server, υποθέτοντας ότι υπάρχει αξιόπιστη σύνδεση με αυτόν. Το λογισμικό θα είναι επίσης υπεύθυνο για την καθοδήγηση του UAV με εντολές Mavlink, χρησιμοποιώντας την προαναφερόμενη σειριακή διεπαφή, προσπαθώντας να διατηρεί τον στόχο στο κέντρο των καρέ.

Όταν λάβει την κατάλληλη οδηγία από τον server ή απευθείας από το σύστημα τηλεχειρισμού σε περίπτωση που λειτουργεί χωρίς σύνδεση με το server, ανιχνεύει αντικείμενα στο πιο πρόσφατο καρέ που διαβάστηκε από την κάμερα και επιλέγει το αντικείμενο που βρίσκεται πιο κοντά στην εκτιμώμενη θέση του στόχου που έχει ληφθεί (για άμεσο τηλεχειρισμό, αυτή η θέση είναι το κέντρο της εικόνας). Στη συνέχεια, προχωρά στο tracking του αντικειμένου στα επόμενα καρέ, και ανα τακτά χρονικά διαστήματα (π.χ. κάθε 5 δευτερόλεπτα) εκτελεί ξανά τη διαδικασία ανίχνευσης και αν βρεθεί ένα αντικείμενο κοντά στην τροχιά του στόχου, ο στόχος ενημερώνεται. Αυτό είναι σημαντικό κυρίως επειδή το αντικείμενο μπορεί να αρχίσει να απομακρύνεται σταδιακά από το παρακολουθούμενο πλαίσιο.

Όταν το πρόγραμμα ανιχνεύει αλλαγή στη λειτουργία πτήσης του αυτόματου πιλότου σε "Guided" (όταν ο χρήστης το κατευθύνει μέσω τηλεχειρισμού, η λειτουργία ρυθμίζεται σε "Loiter" ή "Position Hold"), αναλαμβάνει τον έλεγχο της πλοήγησης του UAV και ρυθμίζει την ταχύτητά του κατάλληλα, ώστε να ακολουθεί τον παρακολουθούμενο στόχο. Όταν η λειτουργία πτήσης επαναφερθεί σε "Position Hold", οι εντολές ρύθμισης ταχύτητας σταματούν να αποστέλλονται.

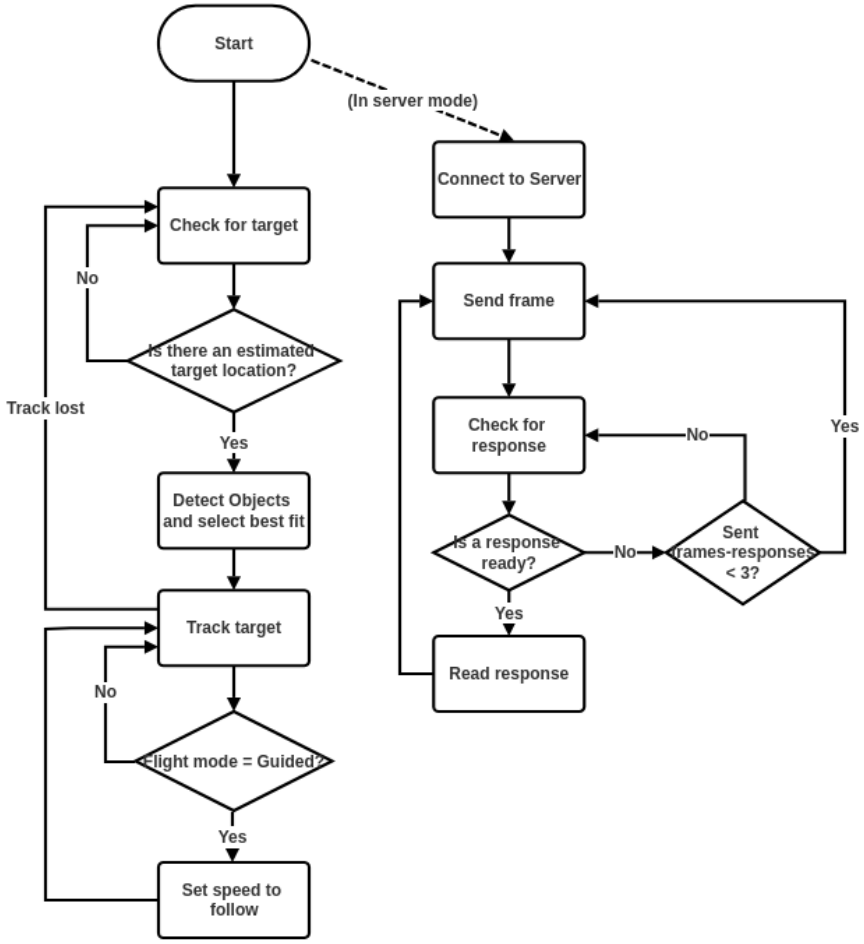


Figure 3.2: Μια επισκόπηση της δομής του λογισμικού του UAV

Η επικοινωνία με τον server πραγματοποιείται σε ένα ξεχωριστό νήμα, το οποίο στέλνει τα πιο πρόσφατα ληφθέντα καρέ με τη μέγιστη δυνατή συχνότητα και περιμένει τις απαντήσεις του server που υποδεικνύουν τη θέση του στόχου. Η διαδικασία λήψης είναι non-blocking, καθώς είναι πιθανόν να χρειαστεί να αποσταλούν νέα καρέ προτού φτάσει η απάντηση που σχετίζεται με τα προηγούμενα στο UAV. Ωστόσο, η διαφορά μεταξύ των αποσταλμένων καρέ και των απαντήσεων είναι περιορισμένη σε 3, προκειμένου να αποφευχθεί η εισαγωγή πρόσθετης καθυστέρησης. Τα καρέ αναλύονται και συμπιέζονται σε μορφή jpg, για να μειωθεί ο όγκος της μεταδιδόμενης πληροφορίας. Αποστέλλονται επίσης μαζί με ένα timestamp, καθώς οι ομοιόμορφες χρονικές αποστάσεις μεταξύ τους δεν είναι εγγυημένες, και ο server θα χρειαστεί αυτή την πληροφορία για να προβλέψει την μελλοντική διαδρομή του οχήματος και να στείλει ακριβείς οδηγίες στο UAV.

3.2 Απομακρυσμένος Server

Ο server που εκτελεί τα πιο βαριά μοντέλα λειτουργεί με Ubuntu 22.04 και είναι εξοπλισμένος με μια GPU NVIDIA A40. Σκοπός του είναι η ανίχνευση οχημάτων με ακριβή μοντέλα, που δεν μπορούν να εκτελεστούν άμεσα στο UAV, η παρακολούθηση των υποδεικνυόμενων στόχων και ο επαναεντοπισμός τους τους σε περίπτωση που είναι απαραίτητο. Η επεξεργασία των καρέ έχει ως αποτέλεσμα μια ακολουθία προβλέψεων μελλοντικών θέσεων που αποστέλλονται στο Raspberry Pi του UAV, το οποίο τις χρησιμοποιεί για να εντοπίσει εκ νέου τον στόχο σε περίπτωση που χαθεί ή είναι λανθασμένος. Παρέχει επίσης μια διεπαφή για αλληλεπίδραση με τον χρήστη, που μπορεί να μεταδίδει βίντεο με επισημειώσεις των ανιχνευμένων τροχιών και να προσδιορίζει τον στόχο που ενδιαφέρει τον χρήστη με βάση το αναγνωριστικό της διαδρομής του.

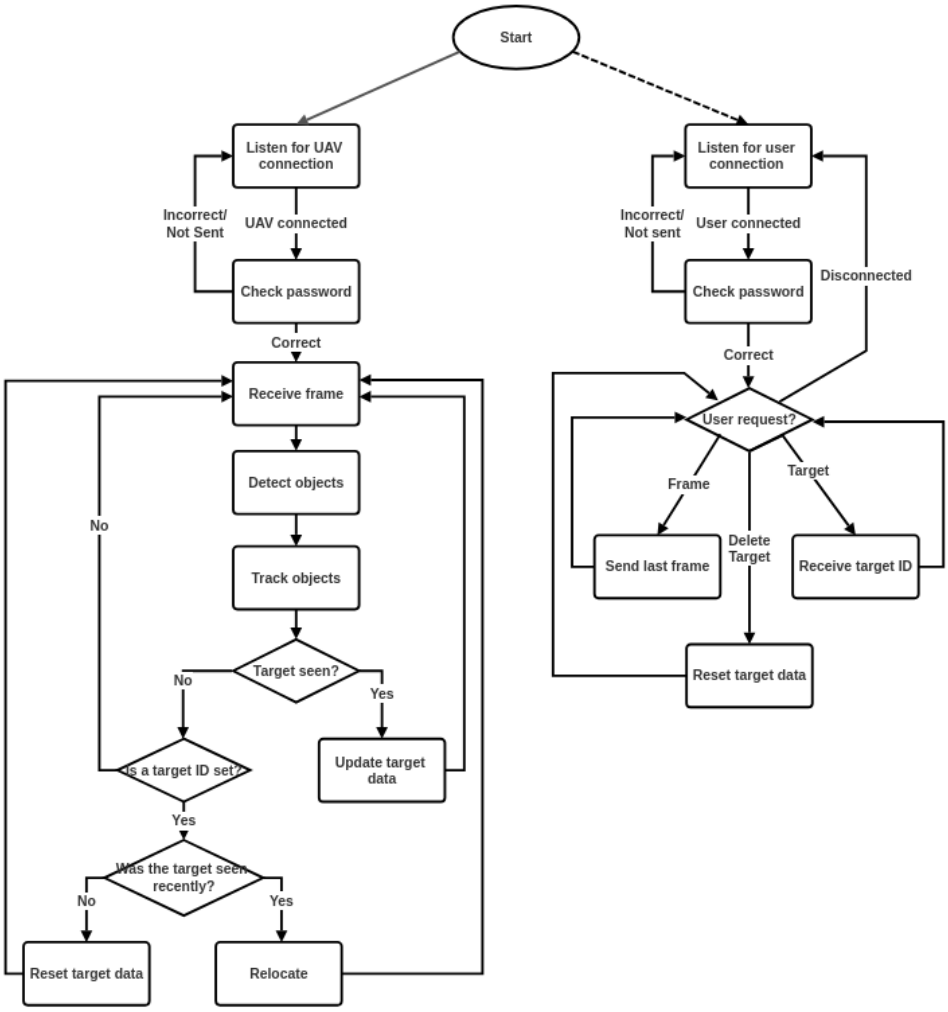


Figure 3.3: Μια επισκόπηση της δομής του λογισμικού του server

3.2.1 Ανίχνευση και Παρακολούθηση Αντικειμένων

Ο server, σε αντίθεση με το Raspberry Pi, εκτελεί τη διαδικασία ανίχνευσης για κάθε καρέ που λαμβάνει, καθώς η GPU του τού επιτρέπει να το κάνει σε πραγματικό χρόνο. Στη συνέχεια, παρακολουθεί πολλά ανιχνευμένα οχήματα χρησιμοποιώντας έναν ανιχνευτή που λαμβάνει υπόψη σύνθετα χαρακτηριστικά των ανιχνευμένων αντικειμένων (Deep Sort tracker).

Όταν ληφθεί ένα νέο καρέ, επαναφέρεται στις αρχικές του διαστάσεις, ώστε τα αποτελέσματα να αντιστοιχούν απευθείας σε «πραγματικές» θέσεις στο καρέ που βλέπει το Raspberry Pi, και η ανίχνευση εκτελείται από ένα μοντέλο YOLOv8 (επιλεγμένο με βάση τα αποτελέσματα που παρουσιάζονται στο κεφάλαιο 6) εκπαιδευμένο σε κατάλληλα δεδομένα, χρησιμοποιώντας την GPU του server. Μαζί με την ανίχνευση των οχημάτων, παράγονται επίσης διανύσματα χαρακτηριστικών για κάθε bounding box.

3.2.2 Πρόβλεψη Διαδρομής και Επαναεντοπισμός Στόχου

Η πρόβλεψη της μελλοντικής διαδρομής του οχήματος-στόχου υλοποιείται χρησιμοποιώντας ένα απλό δίκτυο LSTM και έχει 2 κύριους σκοπούς: Πρώτον, η επικοινωνία μεταξύ του UAV και του server προκαλεί καθυστέρηση μεταξύ της αποστολής ενός καρέ και της λήψης μιας απάντησης, επομένως ο server θα πρέπει να στείλει μια ακολουθία προβλέψεων με timestamps, από τις οποίες το Raspberry θα επιλέξει αυτή που είναι πιο κοντά στην τρέχουσα χρονική στιγμή. Οι προβλέψεις θα έχουν επίσης σημαντική συμβολή στον επαναεντοπισμό ενός χαμένου στόχου, παρέχοντας μια προσεγγιστική εκτίμηση της θέσης του. Η είσοδος σε αυτό το δίκτυο είναι μια ακολουθία από τις προηγούμενες 10 καταγεγραμμένες θέσεις του στόχου, οι οποίες καταγράφονται κατά τη διάρκεια της επεξεργασίας των καρέ όπου ο στόχος είναι ορατός.

Αυτός ο επαναεντοπισμός βασίζεται σε 3 μετρικές που υποδεικνύουν την ομοιότητα ενός ανιχνευμένου αντικειμένου με τον χαμένο στόχο: η τοποθεσία του (σε σύγκριση με τις παραπάνω προβλέψεις), η ομοιότητα των χαρακτηριστικών του με αυτά του στόχου, τα οποία το πρόγραμμα καταγράφει ενώ παρακολουθεί τον στόχο, και οι διαστάσεις του. Η ομοιότητα των χαρακτηριστικών υπολογίζεται ως το εσωτερικό γινόμενο μεταξύ των διανυσμάτων χαρακτηριστικών που παράγονται από τα μοντέλα ανίχνευσης. Η ομοιότητα διαστάσεων είναι πιθανώς εν μέρει κωδικοποιημένη στην ομοιότητα χαρακτηριστικών και δεν θα παρέχει ιδιαίτερα χρήσιμες πληροφορίες σε μια σκηνή με πολλά οχήματα του ίδιου τύπου, επομένως δεν θεωρείται τόσο σημαντική όσο οι άλλες 2 μετρικές.

Κεφάλαιο 4

Σύνολα Δεδομένων και Προεπεξεργασία

4.1 VisDrone

Η εξειδικευμένη εκπαίδευση των χρησιμοποιούμενων μοντέλων ανίχνευσης και πρόβλεψης διαδρομής πραγματοποιήθηκε στα dataset VisDrone2018-DET και VisDrone-VDT2018 αντίστοιχα.

4.1.1 VisDrone2018-DET

Το σύνολο δεδομένων που παρέχεται για την πρόκληση VisDrone2018-DET [44] αποτελείται από ένα σύνολο 8.599 εικόνων από την οπτική ενός UAV, με λεπτομερείς επισημάνσεις, εστιάζοντας σε αυτοκίνητα και πεζούς. Επισημάνσεις άλλων οχημάτων (βαν, λεωφορείο, φορτηγό, μοτοσικλέτα, ποδήλατο, τρίκυκλο) επίσης περιλαμβάνονται, ωστόσο ο αριθμός των εμφανίσεών τους δεν είναι ομοιόμορφος, όπως φαίνεται στο παρακάτω γράφημα.

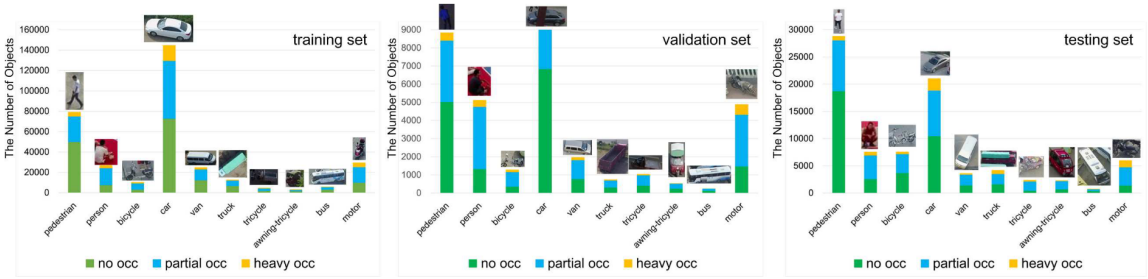


Figure 4.1: Εμφανίσεις αντικειμένων στα υποσύνολα εκπαίδευσης, επικύρωσης και δοκιμής. Επίσης υποδεικνύεται η κάλυψη (occlusion) αντικειμένων (εικόνα από [44])

Το σύνολο δεδομένων παρέχεται χωρισμένο σε 3 υποσύνολα, για εκπαίδευση (6.471 εικόνες),

επικύρωση (548 εικόνες) και δοκιμή (1.580 εικόνες). Η σημασία του συνόλου δεδομένων είναι μεγάλη, καθώς η διαφορά μεταξύ της κάτοψης και πλάγιας όψης του αντικειμένου (συνηθισμένη σε πολλά σύνολα δεδομένων) μπορεί να αποτελέσει μεγάλο εμπόδιο για τη διαδικασία ανίχνευσης.

Το σύνολο δεδομένων περιλαμβάνει εικόνες που έχουν ληφθεί σε διάφορες συνθήκες (για παράδειγμα διαφορετικός φωτισμός, αποστάσεις από τα αντικείμενα, οπτική γωνία, περιβάλλον). Ορισμένα από τα ανιχνευμένα αντικείμενα είναι επίσης μερικώς εντός του οπτικού πεδίου της κάμερας ή επικαλυμμένα. Οι αντίστοιχες επισημάνσεις παρέχουν αυτές τις πληροφορίες μαζί με την κατηγορία και τις πληροφορίες του bounding box.

Figure 4.2: Παραδείγματα επισημάνσεων από το σύνολο δεδομένων. Οι διακεκομμένες γραμμές υποδεικνύουν επικαλυμμένα αντικείμενα (εικόνα από [44])

4.1.2 VisDrone-VDT2018

Το σύνολο δεδομένων για την πρόκληση VisDrone-VDT2018 [45] είναι ένα σύνολο 79 βίντεο κλιπ, με συνολικό αριθμό 33.366 καρέ. Οι επισημασμένες κατηγορίες είναι οι ίδιες με αυτές του ανωτέρω συνόλου δεδομένων ανίχνευσης, και μια παρόμοια ανισορροπία μεταξύ τους είναι παρουσιάζεται και πάλι. Όπως και πριν, τα κλιπ έχουν κινηματογραφηθεί υπό διάφορες συνθήκες, και τα μερικώς επικαλυμμένα ή κομμένα αντικείμενα είναι επισημασμένα. Η κύρια διαφορά με το σύνολο δεδομένων ανίχνευσης είναι ότι το VDT2018 περιλαμβάνει επίσης αναγνωριστικά (IDs) που αποδίδονται σε κάθε ανιχνευμένο αντικείμενο, ώστε να μπορεί να παρακολουθηθεί.

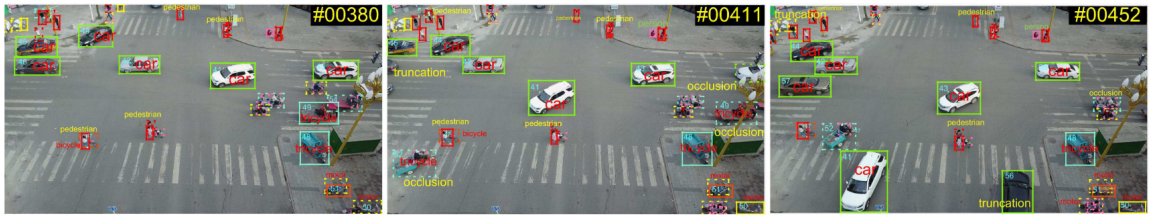


Figure 4.3: Παραδείγματα επισημάνσεων από το σύνολο δεδομένων, μαζί με τα IDs που τους έχουν αποδοθεί σε καρέ ενός κλιπ (εικόνα από [45])

4.2 Απαιτούμενες Μορφές Δεδομένων

Για να επεξεργαστούν την είσοδό τους και να εκπαιδευτούν, τα μοντέλα που θα χρησιμοποιηθούν απαιτούν συγκεκριμένες μορφές δεδομένων που θα πρέπει να προετοιμαστούν είτε με την κατάλληλη διαχείριση των εισόδων κατά την εκτέλεση (στην πρώτη περίπτωση) είτε, στη δεύτερη περίπτωση, με την προεπεξεργασία των διαθέσιμων δεδομένων εκπαίδευσης.

4.2.1 Ανίχνευση

Για την ανίχνευση αντικειμένων, τα δεδομένα εισόδου είναι η εικόνα που θα υποστεί επεξεργασία. Ωστόσο, τα δεδομένα του VisDrone θα χρειαστούν μια ελαφριά τροποποίηση για τη διαδικασία εκπαίδευσης. Η αρχική μορφή των καταγραφών του συνόλου δεδομένων είναι η εξής, όπου left, top, width και height είναι οι συντεταγμένες x,y της επάνω αριστερής γωνίας και το πλάτος και το ύψος του πλαισίου, score είναι ένας δυαδικός δείκτης σχετικός με την πρόκληση VisDrone και τα τελευταία 2 πεδία υποδεικνύουν τη σοβαρότητα της περικοπής και της επικάλυψης:

left	top	width	height	score	category	truncation	occlusion

Για το μοντέλο που εκπαιδεύτηκε για το Raspberry Pi (YOLOv5n), καθώς και για τα άλλα μοντέλα ανίχνευσης της Ultralytics που εκπαιδεύτηκαν (YOLOv8 και RT-DETR), τα δεδομένα επειτα απο επεξεργασία καταλήγουν στη μορφή:

category	x_center	y_center	width	height

όπου οι συντεταγμένες και οι διαστάσεις του πλαισίου κανονικοποιούνται ως προς τις διαστάσεις της εικόνας.

Ορισμένες σπάνιες και λιγότερο σημαντικές κατηγορίες αντικειμένων επαναταξινομήθηκαν και συγχωνεύθηκαν με παρόμοιες για να αποφευχθεί η περιττή δυσκολία εκπαίδευσης.

4.2.2 Πρόβλεψη Διαδρομής

Δεδομένα από το σύνολο VisDrone-VDT2018 χρησιμοποιήθηκαν για την εκπαίδευση του μοντέλου πρόβλεψης διαδρομής. Το LSTM για την πρόβλεψη διαδρομής χρειάζεται να λάβει υπόψη

5 θέσεις σε κάθε χρονικό βήμα: Αυτή του παρακολουθούμενου στόχου και των 4 πλησιέστερων γειτόνων του (πάνω, κάτω, δεξιά, αριστερά). Επομένως, για κάθε χρονικό βήμα θα χρειαστεί να παρασχεθεί ένα διάνυσμα μήκους 10 (2 τιμές συντεταγμένων για κάθε θέση). Το μοντέλο εκπαιδεύτηκε να προβλέπει μια διαδρομή 6 μελλοντικών βημάτων χρησιμοποιώντας ένα παράθυρο 10 προηγούμενων βημάτων.

Τα δεδομένα του VisDrone-VDT2018 παρέχονται σε μορφή παρόμοια με αυτήν που αναφέρθηκε παραπάνω, με τη διαφορά ότι περιλαμβάνονται 2 επιπλέον πεδία, ένα για την επισήμανση της θέσης του καρέ στο βίντεο και ένα για το track ID του αντικειμένου. Για να πραγματοποιηθεί η εκπαίδευση του μοντέλου, οι επισημάνσεις για κάθε βίντεο μετατράπηκαν σε μια λίστα καταχωρήσεων με τα απαραίτητα δεδομένα και έναν δείκτη τροχιάς και αποθηκεύτηκαν. Κατά τη διαδικασία εκπαίδευσης, αυτές οι καταχωρήσεις χωρίστηκαν κατάλληλα σε ομάδες 10+6 βημάτων (προηγούμενα βήματα και στόχοι). Για τη λειτουργία σε πραγματικό χρόνο, τα βήματα που απαιτούνται για την πρόβλεψη συγκεντρώνονται σταδιακά κατά τη διαδικασία παρακολούθησης και μετατρέπονται στην προαναφερθείσα μορφή.

Κεφάλαιο 5

Λογισμικό UAV

5.1 Μοντέλο Ανίχνευσης

5.1.1 Επιλογή Μοντέλου και Αποτελέσματα στο NCS2

Όπως αναφέρθηκε στο Κεφάλαιο 3, το τοπικό μοντέλο στο Raspberry Pi θα πρέπει να εκτελείται στο Intel NCS2. Παρόλο που επιτρέπει την επεξεργασία σχετικά σύνθετων νευρωνικών δικτύων, οι δυνατότητές του είναι περιορισμένες. Επομένως, ένα ελαφρύ και αποτελεσματικό μοντέλο ανίχνευσης θα είναι απαραίτητο για αυτήν την εργασία. Οι πρώτες επιλογές για αυτόν τον σκοπό είναι τα μοντέλα YOLO. Δεδομένου ότι η κυκλοφορία του NCS2 έχει διακοπεί, οι τελευταίες εκδόσεις δεν υποστηρίζονται, και η πιο πρόσφατη που μπορεί να χρησιμοποιηθεί με επιτυχία είναι η YOLOv5. Λόγω υπολογιστικών περιορισμών, χρησιμοποιήθηκε η μικρότερη έκδοση (YOLOv5n).

Η επεξεργασία πραγματοποιήθηκε χρησιμοποιώντας το OpenVINO inference engine. Η αρχική έξοδος που παράγεται από τη συσκευή για το μοντέλο είναι ένα διάνυσμα που περιέχει τα αποτελέσματα για όλα τα πιθανά bounding boxes (κάποιες χιλιάδες), με 85 τιμές που αντιστοιχούν σε κάθε bounding box, υποδεικνύοντας τις συντεταγμένες του (4 τιμές), την πιθανότητα εγκυρότητάς του (1 τιμή) και τις πιθανότητες κατηγοριοποίησης (80 τιμές αν χρησιμοποιούνται οι κατηγορίες COCO). Για να αποκτηθούν τα τελικά αποτελέσματα, επιλέγεται η υψηλότερη πιθανότητα κατηγορίας και τα bounding boxes με πιθανότητα πάνω από ένα καθορισμένο κατώφλι θεωρούνται έγκυρα.

Σε μια προσπάθεια να ελαχιστοποιηθεί ο χρόνος επεξεργασίας, το μοντέλο ρυθμίστηκε να δέχεται εικόνες της χαμηλότερης ανάλυσης που μπορούν να παράγουν αξιόπιστα αποτελέσματα (270x512 εικονοστοιχεία). Με αυτή τη ρύθμιση, ο χρόνος επεξεργασίας για ένα καρέ είναι περίπου 50ms. Το κόστος επεξεργασίας της αρχικής εξόδου είναι πολύ χαμηλό χρησιμοποιώντας vectorized operations (περίπου 2ms). Φυσικά, η διαδικασία ανίχνευσης δεν είναι η μόνη εργασία που πρέπει να εκτελεί το Raspberry Pi, ωστόσο, δεδομένου ότι το βήμα ανίχνευσης δεν επαναλαμβάνεται για κάθε καρέ, η ταχύτητα αυτή επιτρέπει πρακτικά την επεξεργασία βίντεο σε πραγματικό χρόνο.

5.1.2 Εκπαίδευση σε Εξειδικευμένα Δεδομένα

Για να επιτευχθούν καλύτερα αποτελέσματα, ήταν απαραίτητη η εκπαίδευση του μοντέλου σε εξειδικευμένα δεδομένα. Για αυτόν τον σκοπό χρησιμοποιήθηκε το σύνολο δεδομένων VisDrone2018-DET που αναφέρθηκε στο προηγούμενο κεφάλαιο. Τα βάρη που είχαν εκπαιδευτεί προηγουμένως στο σύνολο δεδομένων COCO ρυθμίστηκαν περαιτέρω (fine-tuning), επιτυγχάνοντας σημαντική μείωση των σφαλμάτων και καλύτερα αποτελέσματα, όπως φαίνεται παρακάτω:

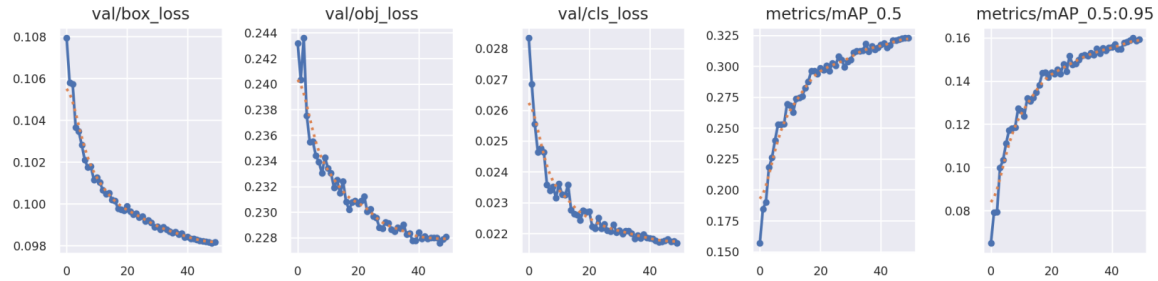


Figure 5.1: Μετρικές απωλειών και σκορ ακρίβειας κατα το fine-tuning του μοντέλου (validation set)

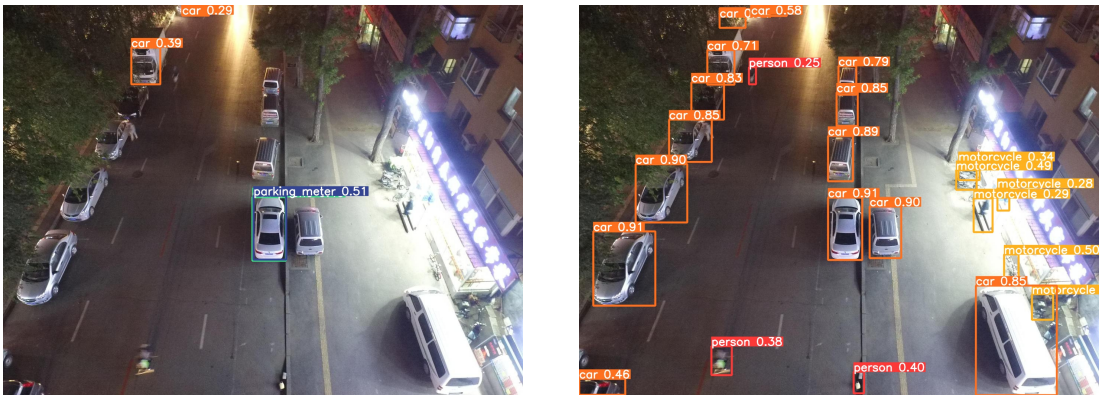


Figure 5.2: Ενδεικτικά αποτελέσματα ανίχνευσης πριν και μετά τη διαδικασία fine-tuning

Λόγω των σημαντικών διαφορών μεταξύ των εικόνων του drone και των εικόνων στο σύνολο δεδομένων COCO, θα μπορούσε να είναι δυνατό να επιτευχθούν παρόμοια ή καλύτερα αποτελέσματα εκπαιδεύοντας το μοντέλο από την αρχή. Μια προσπάθεια για αυτό είχε ως αποτέλεσμα τα παρακάτω:

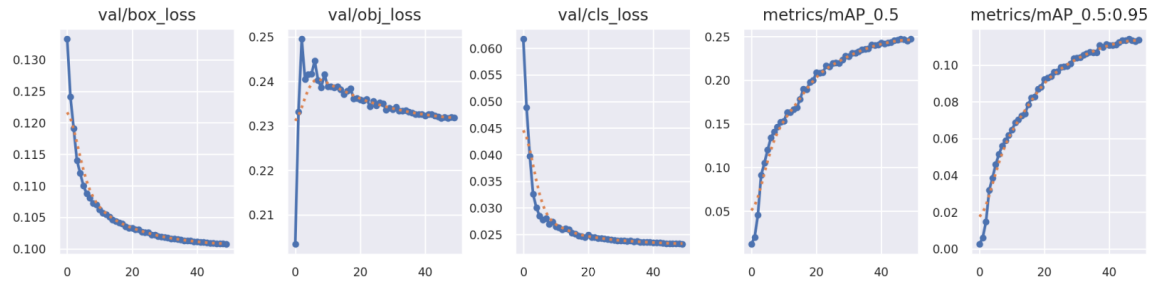


Figure 5.3: Μετρικές απωλειών και σκορ ακρίβειας κατα την εκπαίδευση του μοντέλου από την αρχή (validation set)

Η απόδοση του μοντέλου που έχει προεκπαιδευτεί και έχει τελειοποιηθεί με fine-tuning ήταν ελαφρώς καλύτερη, επομένως ήταν αυτό που επιλέχθηκε στην υλοποίηση του συστήματος.

5.2 Tracking

Για την παρακολούθηση του επιθυμητού στόχου, επιλέχθηκε ο ανιχνευτής KCF που έχει υλοποιηθεί στην OpenCV, καθώς παρέχει επαρκή αξιοπιστία ενώ έχει σχετικά χαμηλό κόστος χρόνου. Το μοντέλο ανίχνευσης παρέχει τις τοποθεσίες των αντικειμένων ενδιαφέροντος στη σκηνή, και αφού βρεθεί η καλύτερη αντιστοιχία, ξεκινά ένα αντικείμενο tracker χρησιμοποιώντας το δοσμένο bounding box. Η διαδικασία ανίχνευσης σταματά για μια καθορισμένη περίοδο (5 δευτερόλεπτα στην εξεταζόμενη υλοποίηση) - αν το παρακολουθούμενο αντικείμενο δεν χαθεί κατά τη διάρκεια αυτού του χρόνου - και στη συνέχεια πραγματοποιείται νέα ανίχνευση, η οποία χρησιμοποιείται για την ενημέρωση του παρακολουθούμενου στόχου αν υπάρχει καλή αντιστοιχία. Αυτό είναι απαραίτητο λόγω αλλαγών που μπορεί να συμβούν στον τρόπο που το αντικείμενο απεικονίζεται στο καρέ (για παράδειγμα σε περίπτωση αλλαγής απόστασης του UAV) ή πιθανής σταδιακής μετατόπισης του bounding box σε σχέση με τη θέση του αντικειμένου.

Figure 5.4: Παράδειγμα περίπτωσης όπου απαιτείται ανανέωση του παρακολουθούμενου bounding box

5.3 Μετάδοση Δεδομένων

Ο προεπιλεγμένος τρόπος επικοινωνίας με τον server είναι μέσω WiFi, υποθέτοντας ότι το UAV λειτουργεί σε μια περιοχή με καλή κάλυψη. Η αξιόπιστη σύνδεση είναι κρίσιμη για να διασφαλιστεί ότι οι σωστές οδηγίες παρέχονται εγκαίρως από τον server στο UAV. Τα καρέ συμπιέζονται σε μορφή jpg και αποστέλλονται στον server μέσω TCP/IP, αφού έχουν μετατραπεί σε εικόνες μικρότερου μεγέθους. Το UAV δεν στέλνει περισσότερα από 3 καρέ στον server χωρίς να λάβει απάντηση.

Ένας πιθανός τρόπος για να αποφευχθεί η ανάγκη για κάλυψη WiFi θα μπορούσε να είναι η χρήση ενός Waveshare 4G HAT για το Raspberry Pi, όπως αναφέρεται στο Κεφάλαιο 3. Το 4G HAT μπορεί να επικοινωνεί με το Raspberry χρησιμοποιώντας εντολές AT. Αυτές που χρησιμοποιήθηκαν στην εξεταζόμενη υλοποίηση φαίνονται στον παρακάτω πίνακα:

Εντολή	Περιγραφή
AT+CPIN=<pin>	Εισαγωγή PIN SIM
AT+NETOPEN	Έναρξη Υπηρεσίας TCP/IP
AT+CIPOPEN=<link, protocol, address, port>	Σύνδεση socket ως Πελάτης
AT+CIPSEND=<link, length>	Αποστολή Δεδομένων
AT+CIPRXGET=<mode>	Λήψη Δεδομένων
AT+CIPCLOSE=<link>	Κλείσιμο socket
AT+NETCLOSE	Διακοπή Υπηρεσίας TCP/IP

Κατά τη διάρκεια των δοκιμών, διαπιστώθηκε ότι το 4G HAT, αν και συνδέεται και επικοινωνεί επιτυχώς με τον server, δεν μπορεί να μεταδώσει τα απαραίτητα δεδομένα με επαρκή ρυθμό, καθιστώντας αδύνατη την άμεση απόκριση. Επομένως, το UAV πρέπει να λειτουργεί σε μια περιοχή όπου είναι δυνατή η πρόσβαση σε δίκτυο WiFi, ή σε αυτόνομη λειτουργία, χωρίς επικοινωνία με τον server.

5.4 Δημιουργία Εντολών MAVLink

Για να καθοδηγήσει το UAV, το Raspberry Pi χρειάζεται να εκτιμήσει την ταχύτητα του στόχου και στη συνέχεια να στείλει τις κατάλληλες εντολές Mavlink 2 στον αυτόματο πιλότο. Μια αρχική εκτίμηση για την ταχύτητα του στόχου (σε σχέση με το UAV) γίνεται υπολογίζοντας τη σχέση μέτρων ανά πίξελ. Υποθέτοντας ότι οι στόχοι του UAV θα είναι αυτοκίνητα, αυτό υπολογίζεται ως $mpp = \frac{average\ expected\ length\ (m)}{average\ side\ length\ (pixels)}$, όπου το μέσο αναμενόμενο μήκος ορίζεται σε 3 και το μέσο πλάτος είναι το μέσο μήκος του ανιχνευμένου bounding box. Χρησιμοποιώντας αυτή τη σχέση, η κίνηση του bounding box στο καρέ μπορεί να μεταφραστεί σε μια προσέγγιση της πραγματικής ταχύτητας του οχήματος. Επιπλέον της ταχύτητας αυτής, το Raspberry Pi προσπαθεί να διορθώσει την εκτός κέντρου θέση του στόχου εκτιμώντας μια επιπλέον ταχύτητα για το UAV που "κεντράρει" τον στόχο εντός ενός καθορισμένου χρονικου διαστήματος. Αυτό ορίστηκε σε 4 δευτερόλεπτα στην εξεταζόμενη υλοποίηση, ωστόσο μπορεί να εξαρτάται από την εκτιμώμενη απόσταση του στόχου από το κέντρο ή από άλλες παραμέτρους.

Ένα σημαντικό ζήτημα που πρέπει να αντιμετωπιστεί είναι η ανακρίβεια 2 υποθέσεων: ότι το μέσο πλάτος οχήματος είναι 3 μέτρα και ότι οι "συντεταγμένες" της κάμερας είναι οι ίδιες με αυτές του UAV. Για καλύτερα αποτελέσματα, πρέπει να ληφθούν υπόψη η περιστροφή της κάμερας και κάποια σφάλματα στον υπολογισμό του mpp. Υποθέτοντας τα 2 συστήματα συντεταγμένων που φαίνονται στο σχήμα παρακάτω, όταν το Raspberry Pi βλέπει μια αλλαγή $\Delta V_{seen} = mpp \cdot \Delta V_{seen\_pixels\_per\_second}$ στην ταχύτητα του στόχου, μια εκτίμηση για την πραγματική ταχύτητα θα είναι:

$$\Delta V x_{est} = \Delta V x_{seen} \cdot cos(\theta) + \Delta V y_{seen} \cdot sin(\theta) \tag{5.1}$$

$$\Delta V y_{est} = \Delta V y_{seen} \cdot cos(\theta) - \Delta V x_{seen} \cdot sin(\theta) \tag{5.2}$$

Το σφάλμα αυτής της εκτίμησης μπορεί να υπολογιστεί συγκρίνοντας τα παραπάνω με το (γνωστό) αποτέλεσμα της προηγούμενης εντολής πλοήγησης που δόθηκε στον αυτόματο πιλότο. Στην εξεταζόμενη υλοποίηση, αυτό το σφάλμα υπολογίστηκε ως ο μέσος όρος 6 μετρήσεων σε 12 καρέ, έτσι ώστε να ελαχιστοποιηθεί η επίδραση των (πιθανώς συχνών) outliers.

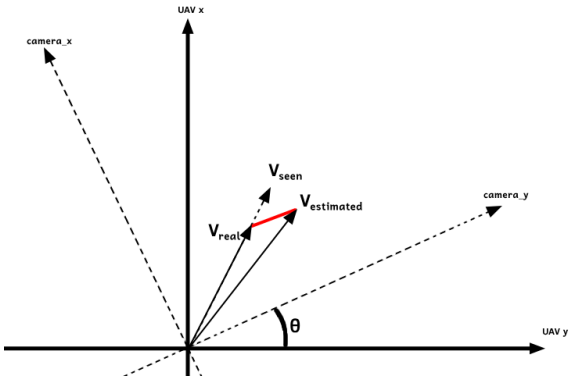


Figure 5.5: Αντιλαμβανόμενη, εκτιμώμενη και πραγματική σχετική ταχύτητα του στόχου. Το σφάλμα σημειώνεται με κοκκινο.

Οι 2 παράμετροι που είναι (κυρίως) υπεύθυνες για αυτό το σφάλμα μπορούν να διορθωθούν μετρώντας την επίδραση μικρών αλλαγών. Αλλαγές που προκαλούν μείωση του σφάλματος ενισχύονται, ενώ εκείνες που προκαλούν αύξηση αντιστρέφονται. Δεδομένης μιας λανθασμένης εκτίμησης της γωνίας θ, οι αλλαγές στο mpp δεν πρόκειται να οδηγήσουν σε σωστή εκτίμηση της παραμέτρου. Ωστόσο, όσο $mpp > 0$, η θ θα μειώνει πάντα το σφάλμα μόνο όταν τροποποιηθεί σωστά. Επομένως, μια εναλλασσόμενη διόρθωση των παραμέτρων είναι βέβαιο οτι θα συγκλίνει στις σωστές τιμές. Για $mpp < 0$, η θ θα συγκλίνει στο $\theta_{real} - \pi$, οδηγώντας το mpp στο $-mpp_{real}$, παράγοντας επίσης σωστά αποτελέσματα.

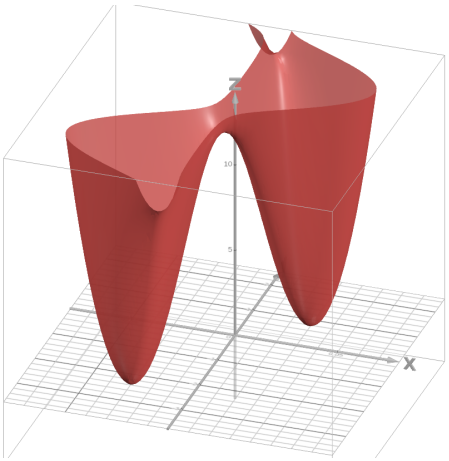


Figure 5.6: Γραφική παράσταση του τετραγωνικού σφάλματος, που κατασκευάστηκε για $Vx_{real} = 3m/s$, $Vy_{real} = 2m/s$, $mpp_{real} = 0.03$ and $\theta_{real} = \frac{\pi}{4}$. Τα ελάχιστα παρατηρούνται στις σχετικές θέσεις(x είναι ο άξονας mpp και y είναι ο άξονας θ)(Η γραφική παράσταση κατασκευάστηκε με το desmos.com)

Κεφάλαιο 6

Απομακρυσμένη Επεξεργασία

6.1 Ανίχνευση Αντικειμένων

Το πρώτο βήμα για την επιτυχή εκτέλεση του tracking είναι φυσικά ένας ακριβής ανιχνευτής αντικειμένων. Για την παρούσα εργασία, όπως συζητήθηκε παραπάνω, θα είναι επίσης απαραίτητο να ληφθούν υπόψη οι χρόνοι επεξεργασίας που μπορούν να επιτευχθούν χρησιμοποιώντας αυτόν τον ανιχνευτή, καθώς απαιτούνται άμεσες αποκρίσεις σε πραγματικό χρόνο. Το διαθέσιμο υλικό στον server επιτρέπει τη χρήση μοντέλων με αυξημένες υπολογιστικές απαιτήσεις με επαρκή ταχύτητα, ωστόσο υπάρχουν ακόμη περιορισμοί που δεν μπορούν να αγνοηθούν.

Σε μια προσπάθεια προσδιορισμού της καλύτερης επιλογής για το μοντέλο ανίχνευσης, δοκιμάστηκαν αρκετοί γνωστοί ανιχνευτές αντικειμένων, οι οποίοι παρατίθενται παρακάτω. Αυτά τα μοντέλα αξιολογήθηκαν με βάση τους χρόνους επεξεργασίας τους, τις επιτυχημένες ανιχνεύσεις τους, την ανοχή τους στη μειωμένη ποιότητα εικόνας – αποδεκτά αποτελέσματα σε χαμηλότερη ποιότητα επιτρέπουν τη μείωση του όγκου των μεταδιδόμενων δεδομένων– και, αργότερα, στην ποιότητα των χαρακτηριστικών που παρέχουν για τα ανιχνευμένα αντικείμενα.

Η εξαγωγή και αξιολόγηση χαρακτηριστικών πρόκειται να είναι καθοριστικής σημασίας για την εργασία του επαναεντοπισμού του στόχου, όπως αναφέρεται στο κεφάλαιο 4. Μπορεί επίσης να εξεταστεί η εναλλακτική εξαγωγής τέτοιων χαρακτηριστικών από τα ανιχνευμένα bounding boxes, είτε χρησιμοποιώντας ένα πολύ απλό συνελικτικό δίκτυο, είτε ένα πιο σύνθετο μοντέλο για βελτίωση της λεπτομέρειας. Ωστόσο, στην πρώτη περίπτωση θα υπάρξει το ζήτημα της χαμηλότερης ποιότητας και (στην περίπτωση πολλών ανιχνευμένων αντικειμένων) πιθανώς σημαντική καθυστέρηση, ενώ στη δεύτερη περίπτωση ο επιπλέον χρόνος επεξεργασίας θα είναι πιθανότατα απαγορευτικός.

Τα αποτελέσματα ανίχνευσης θα αξιολογηθούν κυρίως με βάση έναν προσαρμοσμένο δείκτη ανάκλησης (recall), ο οποίος ορίζεται ως:

$$Recall = \frac{\text{Αριθμός Σωστών Ανιχνεύσεων}}{\text{Αριθμός Σωστών Ανιχνεύσεων + Αριθμός Μη Ανιχνευμένων Αντικειμένων}}$$

Αυτός ο δείκτης επιλέχθηκε λόγω της φύσης του προβλήματος. Οι ψευδώς θετικές ανιχνεύσεις δεν είναι το κύριο ζήτημα, καθώς συγκεκριμένοι στόχοι πρέπει να ανιχνευθούν και να

παρακολουθηθούν, επομένως είναι πιο σημαντικό να "κατανοούνται" αυτοί οι στόχοι ως έγκυρα ανιχνευμένα αντικείμενα. Φυσικά, ένας πολύ μεγάλος αριθμός ψευδών θετικών ανιχνεύσεων μπορεί να προκαλέσει κάποια δυσκολία στον επαναεντοπισμό του στόχου και να προκαλέσει καθυστερήσεις κατά τη διάρκεια της διαδικασίας παρακολούθησης, επομένως θα εξεταστεί και ο δείκτης ακρίβειας (accuracy), ο οποίος ορίζεται ως:

$$Accuracy = \frac{\text{Αριθμός Σωστών Ανιχνεύσεων}}{\text{Αριθμός Σωστών Ανιχνεύσεων + Αριθμός Λανθασμένων Ανιχνεύσεων}}$$

, αν και θα αποτελεί ανησυχία μόνο στην περίπτωση που είναι εξαιρετικά χαμηλός.

Θεωρούμε μια ανίχνευση ως σωστή για ένα αντικείμενο όταν το προβλεπόμενο bounding box έχει αναλογία Intersection over Union πάνω από κάποιο όριο (0.65) με ακριβώς ένα δεδομένο bounding box. Σε περίπτωση πολλαπλών αντιστοιχιών που υπερβαίνουν αυτό το όριο – για παράδειγμα, στην περίπτωση που 2 επισημασμένα αντικείμενα ανιχνεύονται ως ένα – η πρόβλεψη δεν θεωρείται σωστή, καθώς δεν προσδιορίζει σαφώς έναν στόχο και πιθανότατα θα προκαλέσει σύγχυση κατά τη διαδικασία του tracking.

6.1.1 DETR

Ο DETR είναι το πιο υπολογιστικά απαιτητικό μοντέλο που θα δοκιμαστεί. Διαθέτει σημαντική δυνατότητα παραλληλισμού, κάτι που θα μειώσει σημαντικά τον χρόνο επεξεργασίας χρησιμοποιώντας το διαθέσιμο υλικό.

Για τις δοκιμές χρησιμοποιήθηκε το μοντέλο DetrForObjectDetection από το πακέτο python "transformers", υλοποιημένο από την Huggingface. Για τη ρύθμιση των παραμέτρων και της αρχιτεκτονικής του μοντέλου, το πακέτο παρέχει την κλάση DetrConfig. Χρησιμοποιώντας αυτήν, είναι δυνατόν να οριστεί το προτιμώμενο backbone, ο αριθμός ερωτημάτων, ο αριθμός κεφαλών προσοχής και ο αριθμός επιπέδων αποκωδικοποιητών, καθώς και ένας μεγάλος αριθμός άλλων παραμέτρων. Για τις αρχικές δοκιμές χρησιμοποιήθηκαν προεκπαιδευμένα βάρη, με εξαγωγέα χαρακτηριστικών resnet-50, χωρίς τροποποίηση των προεπιλεγμένων ρυθμίσεων.

Μετά από δοκιμή του μοντέλου με διάφορες αναλύσεις εικόνας και κατώφλια εμπιστοσύνης, τα αποτελέσματα ήταν τα εξής:

Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.6	0.17	0.32	80
721x428	0.6	0.15	0.31	73
481x285	0.6	0.10	0.28	78
360x214	0.6	0.08	0.24	71
1442x856	0.4	0.19	0.27	78
721x428	0.4	0.17	0.26	71
481x285	0.4	0.12	0.23	70
360x214	0.4	0.09	0.20	70
1442x856	0.2	0.22	0.21	78
721x428	0.2	0.18	0.20	81
481x285	0.2	0.14	0.18	71
360x214	0.2	0.10	0.16	79

Το Recall φαίνεται σχετικά χαμηλό. Μετά από οπτική επιθεώρηση των αποτελεσμάτων, μπορεί να παρατηρηθεί ότι αυτό οφείλεται κυρίως σε 2 παράγοντες: Πρώτον, το σύνολο δεδομένων είναι εξαιρετικά πυκνά επισημασμένο, με αντικείμενα που βρίσκονται πολύ μακριά για να αναγνωριστούν από το μοντέλο (ή που δεν μας ενδιαφέρουν). Δεύτερον, σε ορισμένες περιπτώσεις, το μοντέλο φαίνεται να μην μπορεί να αναγνωρίσει τα επισημασμένα αντικείμενα, πιθανότατα λόγω της παρατήρησης της σκηνής από μια οπτική γωνία στην οποία το προεκπαιδευμένο μοντέλο δεν έχει εκπαιδευτεί επαρκώς. Παρά τη δυσκολία αυτή, μπορεί να φανεί στα παρακάτω παραδείγματα ότι η απόδοση του DETR δεν είναι απογοητευτική.

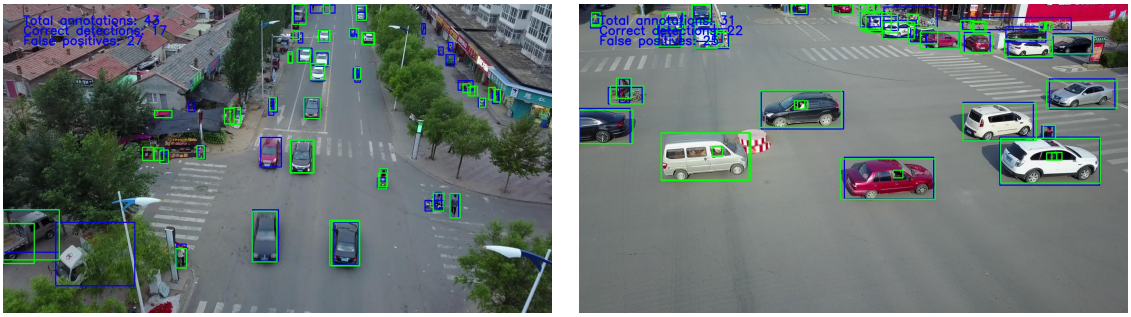


Figure 6.1: Επιτυχημένα αποτελέσματα ανίχνευσης. Τα μπλε πλαίσια αντιπροσωπεύουν τις επισημασμένες αληθείς ανιχνεύσεις και τα πράσινα αντιπροσωπεύουν τις ανιχνεύσεις του μοντέλου.

Εξαγωγή Χαρακτηριστικών

Για την εξαγωγή χαρακτηριστικών που μπορούν ενδεχομένως να συμβάλουν στην επαναεντοπισμό του στόχου, η πιο διαισθητική προσέγγιση είναι η λήψη των διανυσμάτων που παράγει ο αποκωδικοποιητής του transformer ως εξόδους, ακριβώς πριν περάσουν από το πλήρως συνδεδεμένο στρώμα για να παραχθούν τα τελικά bounding boxes.

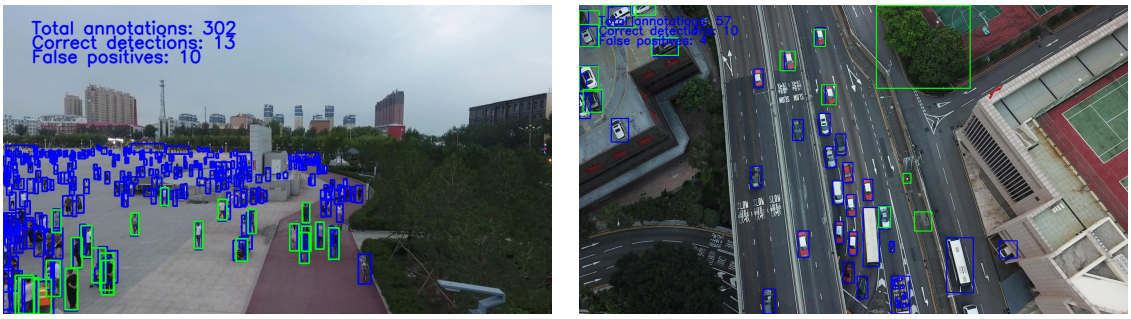


Figure 6.2: Εικόνες με προβληματικές βαθμολογίες (ακραία παραδείγματα). Αριστερά: Πυκνά επισημασμένη εικόνα, με εκατοντάδες πεζούς σε ανοιχτό χώρο. Δεξιά: Η γωνία και πιθανώς το πλαίσιο της εικόνας προκαλούν απώλεια των περισσότερων στόχων.

Σε αυτό το μοντέλο, αυτό μπορεί να γίνει χρησιμοποιώντας το last_hidden_state που επιστρέφεται από τη μέθοδο forward. Οι κρυφές καταστάσεις που επιστρέφονται αντιστοιχούν στα επιστρεφόμενα pred_boxes. Αυτή η αντιστοιχία μπορεί να βοηθήσει στον προσδιορισμό της αντιστοίχισης διανυσμάτων σε bounding boxes. Όπως αναφέρθηκε στο Κεφάλαιο 2, αυτές οι έξοδοι μπορούν να θεωρηθούν ως αποκρίσεις στα ερωτήματα (queries) που χρησιμοποιεί ο αποκωδικοποιητής. Υπάρχει πιθανότητα μεταξύ των καρέ ο στόχος να ανιχνευτεί ως απόκριση σε διαφορετικά ερωτήματα και επομένως σε διαφορετική θέση στην κρυφή κατάσταση. Ακόμα και σε αυτή την περίπτωση, το διάνυσμα μπορεί να χρησιμοποιηθεί για σύγκριση με παλαιότερα διανύσματα χαρακτηριστικών, καθώς τα βάρη του MLP εξόδου είναι κοινά για όλες τις εξόδους του αποκωδικοποιητή.

6.1.2 Deformable DETR

Όπως φαίνεται στο Κεφάλαιο 2, ο μηχανισμός προσοχής του Deformable DETR μπορεί να του επιτρέψει να επιτύχει καλύτερη απόδοση σε μικρά αντικείμενα, κάτι που θα μπορούσε να οδηγήσει σε σημαντικές βελτιώσεις στην παρούσα περίπτωση. Ανεξάρτητα από τις αρχιτεκτονικές διαφορές που εξετάστηκαν στο Κεφάλαιο 2, το μοντέλο που παρέχεται από τους transformers της Huggingface (DeformableDetrForObjectDetection) μπορεί να χρησιμοποιηθεί με τον ίδιο ακριβώς τρόπο όπως το μοντέλο του DETR. Κατά τη διάρκεια της διαδικασίας δοκιμών, διαπιστώθηκε ότι ο Deformable DETR είναι λιγότερο "σίγουρος" από το DETR (για ένα κατώφλι 0.6 χάνει τα περισσότερα οχήματα και πεζούς σχεδόν σε κάθε εικόνα), επομένως δοκιμάστηκε για χαμηλότερα κατώφλια. Με το Resnet-50 ως εξαγωγέα χαρακτηριστικών και με τη χρήση προεκπαιδευμένων βαρών, τα αποτελέσματα ήταν τα εξής:

Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.3	0.18	0.44	168
721x428	0.3	0.16	0.44	163
481x285	0.3	0.11	0.39	162
360x214	0.3	0.08	0.33	162
1442x856	0.2	0.25	0.26	176
721x428	0.2	0.22	0.26	164
481x285	0.2	0.16	0.24	172
360x214	0.2	0.12	0.21	162
1442x856	0.1	0.32	0.13	180
721x428	0.1	0.29	0.13	178
481x285	0.1	0.23	0.11	173
360x214	0.1	0.18	0.09	175

Αυτά τα αποτελέσματα δείχνουν βελτιωμένους δείκτες, αλλά πολύ υψηλότερους χρόνους επεξεργασίας σε σύγκριση με τον DETR. Επίσης, χρησιμοποιώντας το nvidia-smi παρατηρήθηκε χρήση GPU 55-65%, ενώ ο DETR χρησιμοποιούσε μόνο 25-35%. Παρά το γεγονός ότι αυτό το μοντέλο έχει σημαντικά καλύτερες επιδόσεις από τον DETR, δεν μπορεί να χρησιμοποιηθεί σε αυτή την περίπτωση, καθώς υπάρχει ο περιορισμός πραγματικού χρόνου.

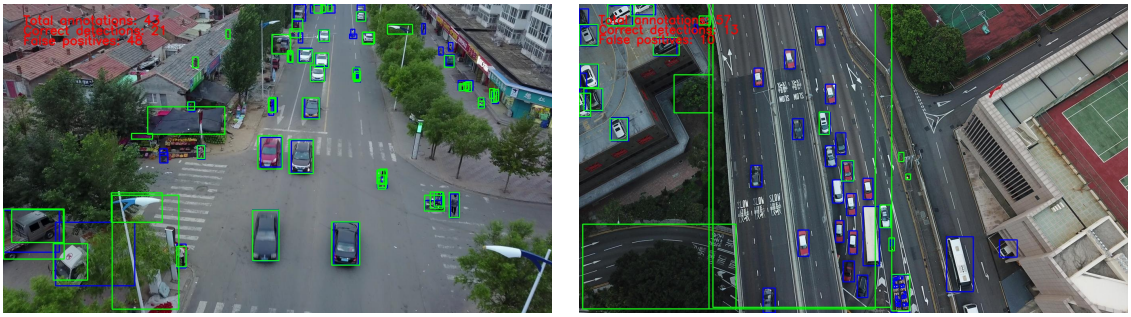


Figure 6.3: Οι εικόνες που φαίνονται στα Σχήματα 6.1-6.2 επεξεργασμένες με το Deformable DETR. Παρά τους υψηλότερους δείκτες, παραμένουν σημαντικές αδυναμίες

6.1.3 RT-DETR

Ο RT-DETR έχει παρουσιάσει αξιοσημείωτα αποτελέσματα και είναι πιθανώς το πιο κατάλληλο μοντέλο της οικογένειας DETR για το συγκεκριμένο σκοπό, καθώς ο σχεδιασμός του έχει βελτιστοποιηθεί τόσο για ακριβή αποτελέσματα όσο και για επεξεργασία σε πραγματικό χρόνο. Η δοκιμή πραγματοποιήθηκε στο μοντέλο RTDETR που υλοποιήθηκε από την ultralytics. Χρησιμοποιήθηκαν βάρη εκπαιδευμένα στο σύνολο δεδομένων COCO και μετρήθηκαν τα εξής αποτελέσματα:

Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.3	0.34	0.52	34.3
481x285	0.3	0.25	0.50	29.4
1442x856	0.2	0.41	0.34	34.8
481x285	0.2	0.32	0.33	30.7
1442x856	0.1	0.49	0.15	34.5
481x285	0.1	0.40	0.15	31.1

Το μοντέλο ανταποκρίνεται καλύτερα και απο τα 2 προηγούμενα, με βελτιωμένο χρόνο υπολογισμού και Recall-Accuracy.

Εξαγωγή Χαρακτηριστικών

Όπως και στην περίπτωση του DETR, θα επιχειρηθεί η εξαγωγή των χαρακτηριστικών των ανιχνευόμενων bounding boxes. Σε αυτή την περίπτωση, δεν θα είναι τόσο απλό, καθώς ο RTDETR της Ultralytics δεν παρέχει τις ακατέργαστες εξόδους του αποκωδικοποιητή. Για να αποκτήσουμε πρόσβαση σε αυτές τις εξόδους, ήταν απαραίτητο να γίνουν μερικές πολύ μικρές τροποποιήσεις στην υλοποίηση του μοντέλου.

Η κλάση RTDETRDecoder που χρησιμοποιείται από το μοντέλο καλεί τη μέθοδο "forward" της κλάσης DeformableTransformerDecoder για να λάβει τα bounding boxes και τις αντίστοιχες κατηγορίες. Σε αυτή τη μέθοδο, οι έξοδοι που παράγονται από τη μέθοδο "forward" της κλάσης DeformableTransformerDecoderLayer περνούν από πλήρως συνδεδεμένα στρώματα για να ληφθεί η τελική εκτίμηση για τις τοποθεσίες των bounding boxes και τα logits κατηγοριών. Στο μοντέλο που χρησιμοποιήθηκε (καθώς και στην αρχιτεκτονική που προτείνεται στο [30]) υπάρχουν 6 στρώματα αποκωδικοποιητή. Για τα χαρακτηριστικά, επιλέχθηκε η έξοδος του τελευταίου στρώματος, ακριβώς πριν περάσει από τα πλήρως συνδεδεμένα στρώματα, και η μέθοδος "forward" της DeformableTransformerDecoder τροποποιήθηκε ώστε να επιστρέφει αυτή την έξοδο μαζί με τα bounding boxes και τις πιθανότητες των κατηγοριών. Υπάρχει μια ακολουθία κλήσεων μεθόδων από άλλες κλάσεις που οδηγεί στην κλήση της προαναφερθείσας μεθόδου "forward", οπότε εισήχθη μια νέα παράμετρος "return_feats", η οποία ορίστηκε σε False από προεπιλογή, καθώς αυτές οι κλάσεις χρησιμοποιούνται και από άλλα μοντέλα.

Όπως και στην περίπτωση του DETR, τα χαρακτηριστικά μπορούν να συγκριθούν μεταξύ τους, καθώς όλα περνούν από το ίδιο δίκτυο για να παραχθούν οι τελικές προβλέψεις. Σε αυτή την περίπτωση, ο αρχικός πίνακας χαρακτηριστικών έχει διαστάσεις [1,300,256], όπου 1 είναι το μέγεθος της παρτίδας, 300 είναι ο αριθμός των επιλεγμένων ερωτημάτων, και 256 είναι το μήκος του διανύσματος χαρακτηριστικών για κάθε bounding box. Τελικά, τα χαρακτηριστικά φιλτράρονται και ο τελικός πίνακας έχει μορφή [nb,256], όπου nb είναι ο αριθμός των bounding boxes για τα οποία η εμπιστοσύνη ανίχνευσης είναι πάνω από κάποιο κατώφλι.

6.1.4 YOLOv8

Τα μοντέλα YOLO φαίνεται να είναι τα πιο αξιόπιστα για το εν λόγω έργο. Οι τελευταίες εκδόσεις, όπως η v8 και η v10, επιτυγχάνουν πολύ υψηλά σκορ mAP σε σύνολα δεδομένων

όπως το COCO, με τις προβλέψεις τους να εκτελούνται σχεδόν σε πραγματικό χρόνο, όπως συζητήθηκε στο Κεφάλαιο 2.

Ακριβώς όπως και για τα προηγούμενα μοντέλα, δοκιμάστηκαν διάφορα μεγέθη εικόνας και κατώφλια εμπιστοσύνης για τις διάφορες εκδόσεις του YOLOv8. Τα αποτελέσματα φαίνονται στους παρακάτω πίνακες:

YOLOv8n				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.03	0.32	9.5
481x285	0.5	0.03	0.29	9.0
1442x856	0.3	0.07	0.39	10.0
481x285	0.3	0.06	0.34	7.2
1442x856	0.1	0.12	0.31	9.8
481x285	0.1	0.10	0.28	7.2
1442x856	0.05	0.15	0.23	9.8
481x285	0.05	0.13	0.22	8.0

YOLOv8m				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.12	0.61	11.6
481x285	0.5	0.08	0.49	8.4
1442x856	0.3	0.18	0.63	11.2
481x285	0.3	0.13	0.53	8.8
1442x856	0.1	0.27	0.47	11.2
481x285	0.1	0.19	0.44	8.6
1442x856	0.05	0.32	0.37	11.6
481x285	0.05	0.23	0.35	8.8

YOLOv8x				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.15	0.69	14.2
481x285	0.5	0.09	0.55	11.8
1442x856	0.3	0.22	0.67	14.0
481x285	0.3	0.15	0.58	12.0
1442x856	0.1	0.31	0.51	14.2
481x285	0.1	0.23	0.47	12.0
1442x856	0.05	0.36	0.41	14.4
481x285	0.05	0.26	0.38	12.0

Η καθυστέρηση και οι μετρικές έχουν βελτιωθεί σημαντικά σε σύγκριση με τα μοντέλα DETR, με εξαίρεση τον RT-DETR, ο οποίος επίσης αποδίδει καλά, αλλά έχει υψηλότερους

χρόνους υπολογισμού. Όπως αναμενόταν, παραμένουν κάποιες δυσκολίες στην ανίχνευση αντικειμένων σε εικόνες με ασυνήθιστη οπτική γωνία, ωστόσο τα αποτελέσματα στις περισσότερες περιπτώσεις είναι ικανοποιητικά, όπως στα παρακάτω παραδείγματα:

Figure 6.4: Παραδείγματα επιτυχημένης ανίχνευσης

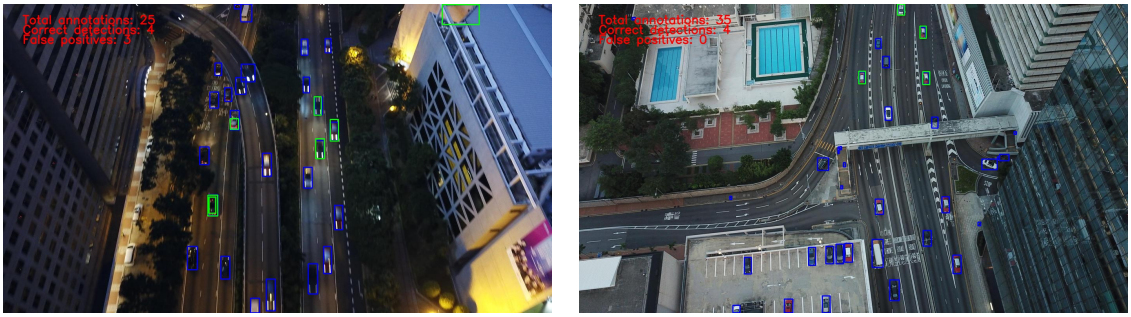


Figure 6.5: Υπάρχουν ακόμη ορισμένες (πολύ λιγότερες) περιπτώσεις όπου παρατηρείται σημαντική δυσκολία.

Εξαγωγή Χαρακτηριστικών

Στην περίπτωση του YOLO, δεν υπάρχει άμεση αντιστοιχία μεταξύ των διανυσμάτων χαρακτηριστικών και των ανιχνευμένων bounding boxes. Είναι, ωστόσο, δυνατό να χρησιμοποιηθούν τα logits κατηγοριών που παράγει το μοντέλο. Για να γίνει αυτό, έπρεπε να γίνουν κάποιες μικρές τροποποιήσεις στην υλοποίηση του μοντέλου YOLOv8.

Σε αυτή την περίπτωση, το μοντέλο παράγει τα συγκεκριμένα logits και κάνει την τελική επιλογή της πιο «πιθανής» κατηγορίας μαζί με τη διαδικασία non-max suppression, οπότε η παρεχόμενη συνάρτηση non_max_suppression() τροποποιήθηκε ώστε να τα επιστρέφει επιπλέον της κανονικής εξόδου της. Παρομοίως με την περίπτωση του RT-DETR, η συνάρτηση καλείται μέσω μιας σειράς κλήσεων μεθόδων από διάφορες κλάσεις που χρησιμοποιούνται σε άλλα μοντέλα, οπότε δημιουργήθηκε μια νέα παράμετρος "return_class_logits" που προεπιλέγεται σε False.

Αυτή τη φορά, ο πίνακας χαρακτηριστικών έχει σχήμα [nb, nc], όπου nb είναι ο αριθμός των

ανιχνεύσεων των οποίων το σκορ εμπιστοσύνης υπερβαίνει το καθορισμένο κατώφλι και έχουν παραμείνει μετά τη διαδικασία non-max suppression, και nc είναι ο αριθμός των κατηγοριών.

Πρέπει να σημειωθεί ότι για το YOLO, δεδομένου ότι αποκτώνται τα logits κατηγοριών αντί για ένα πραγματικό διάνυσμα χαρακτηριστικών, τα αντικείμενα που προβλέπεται ότι ανήκουν στην ίδια κατηγορία με υψηλή βεβαιότητα (όπως συμβαίνει συνήθως στις περιπτώσεις που εξετάζονται) θα φαίνονται ταυτόσημα (θα έχουν ομοιότητα 1). Υπάρχει επίσης το ζήτημα ότι οι κατηγορίες που εμφανίζονται πιο συχνά στις εικόνες - και τις οποίες το μοντέλο έχει εκπαιδευτεί συγκεκριμένα να ανιχνεύει - θα έχουν πολύ υψηλότερες πιθανότητες από τις υπόλοιπες. Για να λυθούν αυτά τα 2 προβλήματα, οι πρώτες 8 κατηγορίες των διανυσμάτων αφαιρέθηκαν. Διαπιστώθηκε επίσης ότι τα αποτελέσματα ήταν ελαφρώς καλύτερα αν η μέγιστη τιμή του εναπομείναντος διανύσματος κλιμακωνόταν σε το πολύ 1.5 φορές την τιμή της δεύτερης μέγιστης, ώστε να αποφευχθεί η υπερβολική συνεισφορά μιας συγκεκριμένης κατηγορίας στο τελικό αποτέλεσμα.

6.1.5 YOLOv10

Για το YOLOv10, η διαδικασία που ακολουθήθηκε και οι λεπτομέρειες της ενσωμάτωσής του στο σύστημα είναι ακριβώς οι ίδιες με αυτές του YOLOv8. Λαμβάνοντας υπόψη τα παραπάνω αποτελέσματα, ειδικά τους πολύ μικρούς χρόνους εκτίμησης, οι δοκιμές πραγματοποιήθηκαν μόνο για την μεγαλύτερη έκδοση του:

YOLOv10x				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.12	0.67	16.2
481x285	0.5	0.07	0.49	13.2
1442x856	0.3	0.18	0.69	15.8
481x285	0.3	0.12	0.55	14.0
1442x856	0.1	0.28	0.56	16.2
481x285	0.1	0.19	0.49	13.6
1442x856	0.05	0.33	0.44	16.0
481x285	0.05	0.23	0.41	13.4
1442x856	0.03	0.36	0.36	16.2
481x285	0.03	0.25	0.35	13.6

Τα αποτελέσματα είναι πολύ παρόμοια (ελαφρώς χειρότερα) με αυτά που επιτεύχθηκαν με το YOLOv8.

6.1.6 Faster R-CNN

Δοκιμάστηκαν επίσης τα Faster RCNN ResNet50 FPN και Faster RCNN ResNet50 FPN v2, όπως υλοποιούνται στο πακέτο python torchvision. Για μια αρχική προσπάθεια, χρησιμοποιήθηκαν τα βάρη που παρέχονται ως προεπιλογή για το μοντέλο, εκπαιδευμένα στο σύνολο δεδομένων COCO.

Faster RCNN ResNet50 FPN				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.18	0.44	71.4
481x285	0.5	0.13	0.40	67.0
1442x856	0.3	0.22	0.34	34.2
481x285	0.3	0.16	0.31	33.0
1442x856	0.1	0.28	0.21	34.0
481x285	0.1	0.20	0.18	32.8

Faster RCNN ResNet50 FPN v2				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.29	0.50	41.0
481x285	0.5	0.15	0.43	40.2
1442x856	0.3	0.33	0.39	41.2
481x285	0.3	0.18	0.35	40.2
1442x856	0.1	0.39	0.25	41.4
481x285	0.1	0.22	0.22	40.4

Όπως φαίνεται παραπάνω, το μοντέλο αποδίδει αρκετά καλά. Ωστόσο, παρόλο που το Faster RCNN ResNet50 FPN v2 έχει σε ορισμένες περιπτώσεις συγκρίσιμες μετρικές με το YOLOv8, έχει σημαντικά υψηλότερη καθυστέρηση. Για τις παραπάνω δοκιμές, παρατηρήθηκε χρήση GPU περίπου 40%. Τα κατώφλια εμπιστοσύνης <0.1 δεν δοκιμάστηκαν, καθώς η ακρίβεια του μοντέλου ήταν σχετικά χαμηλή.

6.1.7 RetinaNet

Η ικανότητα του RetinaNet να ανιχνεύει με επιτυχία πυκνά τοποθετημένα και μικρά αντικείμενα το καθιστά μια πολλά υποσχόμενη επιλογή για την συγκεκριμένη εφαρμογή. Και πάλι, επιλέχθηκαν τα retinanet_resnet50_fpn και retinanet_resnet50_fpn_v2 που υλοποιούνται στο πακέτο torchvision. Τα μοντέλα χρησιμοποιούν ένα backbone ResNet-50-FPN. Για τη δοκιμή, χρησιμοποιήθηκαν βάρη εκπαιδευμένα στο COCO.

RetinaNet ResNet50 FPN				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.08	0.45	35.0
481x285	0.5	0.05	0.36	34.2
1442x856	0.3	0.16	0.31	35.6
481x285	0.3	0.10	0.29	34.2
1442x856	0.1	0.28	0.09	35.6
481x285	0.1	0.19	0.09	34.2

RetinaNet ResNet50 FPN v2				
Μέγεθος Εικόνας	Κατώφλι Εμπιστοσύνης	Recall	Accuracy	Χρόνος (ms)
1442x856	0.5	0.10	0.63	33.4
481x285	0.5	0.05	0.43	32.2
1442x856	0.3	0.24	0.44	33.2
481x285	0.3	0.13	0.38	32.0
1442x856	0.1	0.39	0.12	33.4
481x285	0.1	0.25	0.11	32.2

Και πάλι, τα παραπάνω αποτελέσματα δεν μπορούν να φτάσουν αυτά του YOLOv8. Για confidence_threshold=0.1, το δεύτερο μοντέλο έχει σκορ recall 0.39, ωστόσο η αντίστοιχη ακρίβεια είναι πολύ χαμηλή για να θεωρηθεί αξιόπιστη.

6.2 Εκπαίδευση στο VisDrone2019-DET

Όπως συζητήθηκε, η ασυνήθιστη οπτική γωνία των αντικειμένων που πρέπει να ανιχνευθούν θα έχει πιθανότατα αρνητική επίδραση στα μοντέλα που εκπαιδεύτηκαν σε ένα σύνολο δεδομένων όπως το COCO που δεν περιέχει αρκετά παρόμοια δείγματα, και επομένως απαιτείται fine-tuning σε ένα εξειδικευμένο σύνολο δεδομένων (VisDrone2019-DET). Το validation set που χρησιμοποιήθηκε ήταν σχεδόν 0.1 του συνολικού δείγματος. Επιλέχθηκαν τα καλύτερα μοντέλα από τις "οικογένειες" των συνελικτικών μοντέλων και των μοντέλων τύπου DETR - YOLOv8x και RT-DETR αντίστοιχα.

6.2.1 YOLOv8x

Για το YOLOv8x χρησιμοποιήθηκε η μέθοδος "train" που είναι ενσωματωμένη στο μοντέλο που υλοποιήθηκε απο την Ultralytics. Η διαδικασία εκπαίδευσης πραγματοποιήθηκε με το υλικό του server. Το μοντέλο εκπαιδεύτηκε για 35 εποχές (παρατηρήθηκε ότι επιπλέον εκπαίδευση προκάλεσε αύξηση στις απώλειες στο validation set λόγω overfitting) και η διαδικασία διήρκεσε περίπου 2 ώρες (με χρήση GPU περίπου 95%).

Στον παρακάτω πίνακα, τα αποτελέσματα του νέου μοντέλου μπορούν να συγκριθούν με εκείνα του προεκπαιδευμένου μοντέλου. Οι χρόνοι υπολογισμού δεν περιλαμβάνονται, καθώς είναι πρακτικά πανομοιότυποι.

YOLOv8x					
Image Size	Κατώφλι Εμπιστοσύνης	Pretrained Recall	Pretrained Accuracy	Finetuned Recall	Finetuned Accuracy
1442x856	0.5	0.15	0.69	0.49	0.85
481x285	0.5	0.09	0.55	0.42	0.84
1442x856	0.3	0.22	0.67	0.53	0.74
481x285	0.3	0.15	0.58	0.46	0.73
1442x856	0.1	0.31	0.51	0.58	0.55
481x285	0.1	0.23	0.47	0.50	0.57
1442x856	0.05	0.36	0.41	0.59	0.45
481x285	0.05	0.26	0.38	0.52	0.46

Η βελτίωση είναι προφανής και στους δύο δείκτες. Η πρόοδος της εκπαίδευσης μπορεί να φανεί στα παρακάτω γραφήματα:

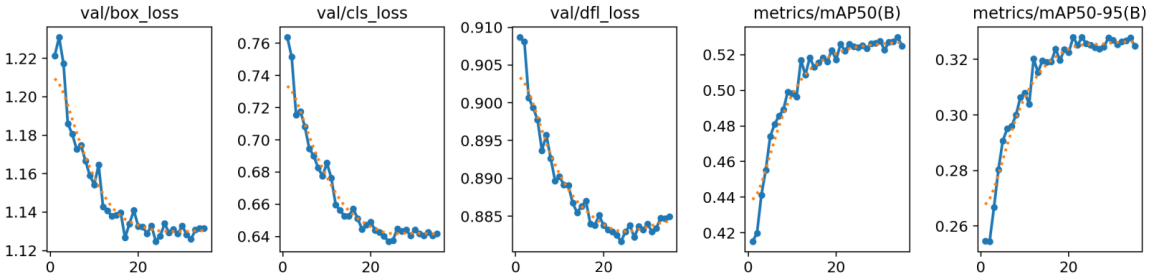


Figure 6.6: Αποτελέσματα απο τη διαδικασία εκπαίδευσης του YOLOv8x (validation set)

Figure 6.7: Αριστερά, πολλές επιτυχείς ανιχνεύσεις ϑολών οχημάτων στο παρασκήνιο, που δεν είχαν καν επισημανθεί στο dataset. Δεξιά, μία από τις εικόνες που προκαλούσαν προβλήματα σε όλα τα μοντέλα που είχαν εκπαιδευτεί στο COCO.

6.2.2 RT-DETR

Όπως προηγουμένως, χρησιμοποιώντας τη διαθέσιμη μέθοδο "train", το μοντέλο προσαρμόστηκε στο σύνολο δεδομένων VisDrone2018-DET για 50 εποχές (τότε οι απώλειες στο

validation set και η ακρίβεια σταθεροποιήθηκαν), και προέκυψαν τα εξής αποτελέσματα:

RT-DETR					
Image Size	Κατώφλι Εμπιστοσύνης	Pretrained Recall	Pretrained Accuracy	Finetuned Recall	Finetuned Accuracy
1442x856	0.3	0.34	0.52	0.60	0.57
481x285	0.3	0.25	0.50	0.50	0.60
1442x856	0.2	0.41	0.34	0.63	0.39
481x285	0.2	0.32	0.33	0.53	0.43
1442x856	0.1	0.49	0.15	0.66	0.19
481x285	0.1	0.40	0.15	0.57	0.20

Όπως και πριν, οι χρόνοι εκπαίδευσης δεν συμπεριλήφθηκαν, καθώς δεν παρατηρήθηκε καμία σημαντική διαφορά. Μπορεί να φανεί ότι τα αποτελέσματα του νέου μοντέλου έχουν βελτιωθεί σημαντικά, και οι επιδόσεις του είναι παρόμοιες με αυτές του YOLOv8 έπειτα απο το fine-tuning όσον αφορά τις μετρικές Recall και Accuracy.

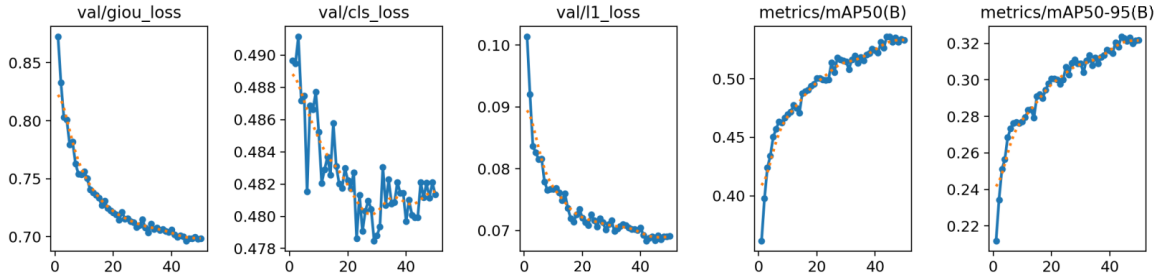


Figure 6.8: Αποτελέσματα που δείχνουν την πρόοδο της εκπαίδευσης του RT-DETR (validation set)

6.3 Πρόβλεψη Διαδρομής

Η διαδικασία της πρόβλεψης της διαδρομής ενός κινούμενου οχήματος με υψηλή ακρίβεια είναι εξαιρετικά περίπλοκη. Προϋποθέτει την καλή γνώση και ικανότητα πρόβλεψης πολλών παραγόντων, συμπεριλαμβανομένων των αλληλεπιδράσεων με εμπόδια, άλλα οχήματα, τις προθέσεις του οδηγού κ.λπ. Για τους σκοπούς της συγκεκριμένης εργασίας, θα επιχειρηθεί μια πολύ απλούστερη (και λιγότερο ακριβής) πρόβλεψη, η οποία θα παρέχει χρήσιμες πληροφορίες για τον επαναεντοπισμό του στόχου όταν είναι απαραίτητο και πιθανώς για τον καθορισμό των μελλοντικών κινήσεων του UAV.

Όπως αναφέρθηκε στο Κεφάλαιο 3, το επιλεγμένο μοντέλο είναι ένα δίκτυο LSTM που ακολουθείται από ένα πλήρως συνδεδεμένο στρώμα που προσπαθεί να προβλέψει τη θέση του στόχου στα επόμενα καρέ. Με αυτόν τον τρόπο, το μοντέλο πρακτικά χρειάζεται να προβλέψει τη συνδυασμένη κίνηση του UAV και του στόχου. Οι κύριοι λόγοι για την επιλογή ενός τέτοιου

μοντέλου ήταν η απλότητα και η ταχύτητα. Μια ενδεχομένως πιο ακριβής και περίπλοκη εναλλακτική θα μπορούσε να περιλαμβάνει την ανίχνευση ακινητοποιημένων αντικειμένων ή άλλων σταθερών σημείων στην σκηνή, την αντιστοίχιση τους μεταξύ των καρέ και τον υπολογισμό της απόλυτης κίνησης του στόχου βάσει αυτής της αντιστοίχισης.

Για αυτήν την εφαρμογή, το μοντέλο έχει ρυθμιστεί να προβλέπει 6 βήματα στο μέλλον. Αυτό συμβαίνει γιατί ο server δεν πρόκειται να λαμβάνει κάθε καρέ από τη ροή βίντεο, και επομένως ο χρόνος μεταξύ των καρέ θα είναι αρκετά μεγάλος, καθιστώντας τις πιο μακροχρόνιες (σε όρους καρέ) προβλέψεις άχρηστες, ειδικά απο τη στιγμή που εκτελούνται από ένα σχετικά ανακριβές μοντέλο και με περιορισμένα δεδομένα. Για τη διαδικασία σε πραγματικό χρόνο θα είναι επίσης απαραίτητο να αντιμετωπιστεί η ανομοιομορφία των χρονικών διαστημάτων μεταξύ των καρέ που θα λαμβάνονται, κάτι που θα επηρεάσει πιθανώς αρνητικά το μοντέλο. Επιπλέον, θα μπορούσε να χρειαστεί να γίνουν προβλέψεις με ατελείς ακολουθίες (όπως αναφέρθηκε στο Κεφάλαιο 5, το μοντέλο αναμένει ακολουθίες 10 προηγούμενων βημάτων, που ενδέχεται να μην είναι διαθέσιμα). Για να αντιμετωπιστούν αυτά τα ζητήματα, θα πρέπει να εφαρμοστεί μια συνάρτηση παρεμβολής-επέκτασης που θα παράγει μια ομοιόμορφα κατανεμημένη ακολουθία 10 βημάτων από τα διαθέσιμα δεδομένα, εφόσον υπάρχουν 2 ή περισσότερα βήματα. Το χρονικό βήμα που παράγει αυτή η συνάρτηση εκτιμάται ως ο μέσος χρόνος μεταξύ της λήψης 2 συνεχόμενων καρέ.

6.3.1 Εκπαίδευση

Η εκπαίδευση του μοντέλου πραγματοποιήθηκε στο dataset VisDrone2019-MOT, με την προετοιμασία των επισημάνσεων που παρουσιάστηκε στο Κεφάλαιο 4. Για τη διαδικασία εκπαίδευσης χρησιμοποιήθηκε ο Adam optimizer που περιλαμβάνεται στο torch.optim, μαζί με μια συνάρτηση απώλειας Μέσου Τετραγωνικού Σφάλματος όπως υλοποιείται στο πακέτο torch.nn (torch.nn.MSELoss()). Διεξήχθησαν αρκετά πειράματα με τις υπόλοιπες υπερπαραμέτρους εκπαίδευσης και τα δεδομένα που δόθηκαν στο μοντέλο.

Συγκεκριμένα, δοκιμάστηκαν διάφορα μεγέθη παρτίδας (16, 32, 64), ρυθμοί εκμάθησης (0.001, 0.0005, 0.0002, 2e-5, 1e-5, 5e-6 και μείωση από το 0.001 έως το 1e-5), κανονικοποιημένα και μη δεδομένα και διαφορετικές πληροφορίες σχετικά με τους γείτονες (χωρίς γείτονες, κοντινότεροι 4 γείτονες όπως περιγράφεται στο Κεφάλαιο 3, κοντινότεροι 4 γείτονες με περιορισμό απόστασης, 12 γείτονες). Το μέγεθος του validation set ήταν σε όλες τις περιπτώσεις 0.2 του συνόλου, και το μοντέλο εκπαιδεύτηκε για 150 εποχές (100 εποχές για την περίπτωση κανονικοποιημένων δεδομένων). Βρέθηκαν τα παρακάτω:

- **Αριθμός γειτόνων:** Οι διάφορες εναλλακτικές που περιγράφηκαν παραπάνω δεν προκάλεσαν σημαντική διαφορά, με εξαίρεση την πλήρη έλλειψη πληροφοριών γειτόνων, που είχε ως αποτέλεσμα ελαφρώς χειρότερα αποτελέσματα (MSE>8 στο σύνολο επικύρωσης με φθίνοντα ρυθμό εκμάθησης, batch_size=32 και κανονικοποιημένα δεδομένα, ενώ οι άλλες ρυθμίσεις πέτυχαν MSE περίπου 6.5-7).
- **Μέγεθος παρτίδας:** Τα μεγέθη παρτίδας που δοκιμάστηκαν δεν φάνηκαν να έχουν σημαντική επίδραση στα τελικά αποτελέσματα. Για μέγεθος παρτίδας 32, φάνηκε ότι η σύγκλιση επιτεύχθηκε μερικές εποχές νωρίτερα.

- **Κανονικοποίηση Δεδομένων:** Η κανονικοποίηση των δεδομένων προκάλεσε πολύ σημαντική βελτίωση στα αποτελέσματα. Χρησιμοποιήθηκε κανονικοποίηση minmax, η οποία προκάλεσε την πτώση του MSE από περίπου 27 σε 13 με learning_rate=0.0001 και μέγεθος παρτίδας 32.

- **Ρυθμός Εκμάθησης:** Η ρύθμιση του ρυθμού εκμάθησης ήταν επίσης καίριας σημασίας. Με ρυθμούς εκμάθησης άνω του 0.0005, το μοντέλο δεν μπορούσε να συγκλίνει σε μια σταθερή κατάσταση, και με πολύ χαμηλούς ρυθμούς εκμάθησης συγκλίνει σε τοπικά ελάχιστα, οδηγώντας σε αυξημένα σφάλματα. Η λύση σε αυτό ήταν η εισαγωγή ενός φθίνοντα ρυθμού εκμάθησης, ακολουθώντας το μοτίβο που φαίνεται παρακάτω, το οποίο είχε ως αποτέλεσμα τελική απώλεια επικύρωσης περίπου 6.5-7 με batch_size=32, 4 γείτονες και κανονικοποιημένα δεδομένα.

	Ep. 0-1	Ep. 2-6	Ep. 7-14	Ep. 15-34	Ep. 35-69	Ep. 70-100
LR	0.005	0.001	0.0005	0.0001	5e-5	1e-5

Κάποια απο τα παραπάνω αποτελέσματα φαίνονται στα γραφήματα που ακολουθούν:

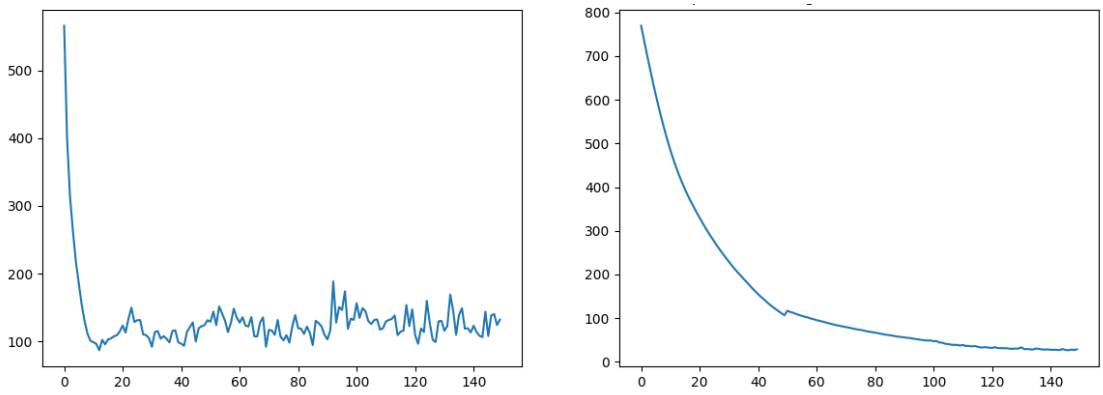


Figure 6.9: Και στις δύο περιπτώσεις χρησιμοποιήθηκαν μη κανονικοποιημένα δεδομένα και BatchSize=32. Αριστερά, η εκπαίδευση πραγματοποιήθηκε με LearningRate=0.001, το οποίο προκαλεί αδυναμία σύγκλισης, και δεξιά, ο ρυθμός εκμάθησης ορίστηκε στο 0.0001, και το μοντέλο συγκλίνει. Ωστόσο, λόγω της έλλειψης κανονικοποίησης, το τελικό MSE είναι κοντά στο 27.

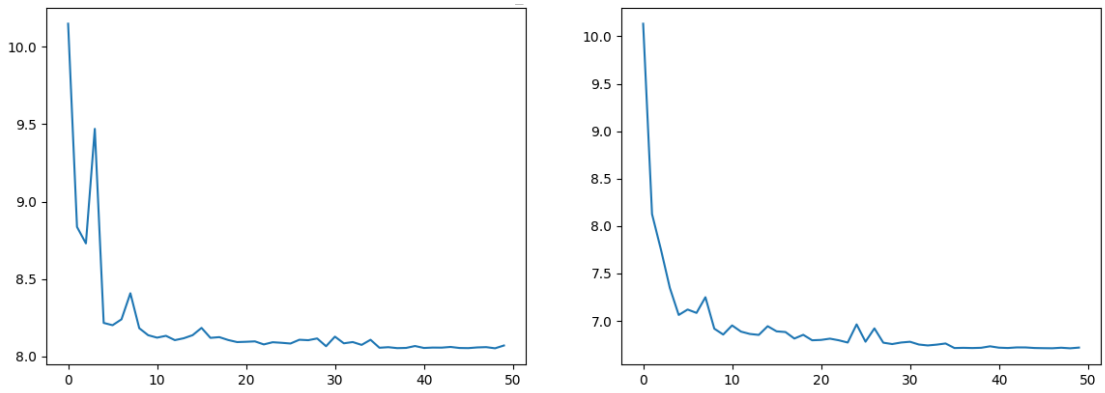


Figure 6.10: MSE με κανονικοποιημένα δεδομένα, BatchSize=32 και φθίνοντα ρυθμό εκμάθησης. Αριστερά υπάρχουν τα αποτελέσματα χωρίς πληροφορίες γειτόνων, και δεξιά τα αποτελέσματα για 4 γείτονες.

6.3.2 Σύγκριση με απλή μεθοδολογία χωρίς μηχανική μάθηση

Ένας τρόπος για να αξιολογηθεί η αποτελεσματικότητα του μοντέλου είναι να συγκριθεί με μια απλή μέθοδο χωρίς τη χρήση κάποιου μοντέλου μηχανικής μάθησης για την εκτίμηση της τροχιάς. Αυτό μπορεί να γίνει με μια προσεγγιστική επέκταση της πορείας που υποδεικνύεται από τα τελευταία σημεία της ακολουθίας βημάτων.

Πριν από αυτό, πρέπει να σημειωθεί ότι τα παραπάνω σφάλματα μπορεί να είναι παραπλανητικά, αν και σίγουρα υποδεικνύουν ότι το μοντέλο μαθαίνει και μπορούν να βοηθήσουν στην επιλογή υπερπαραμέτρων. Αυτό συμβαίνει διότι οι τροχιές επικύρωσης προέρχονται από τα ίδια βίντεο με αυτές που χρησιμοποιήθηκαν κατα την εκπαίδευση. Αν και δεν είναι ίδιες, έχουν καταγραφεί από την ίδια κάμερα που κινείται με τρόπο στον οποίο το μοντέλο μπορεί να έχει εκπαιδευτεί, και σε ορισμένες περιπτώσεις παράγονται από τα ίδια οχήματα με αυτά της εκπαίδευσης σε διαφορετική χρονική στιγμή. Ως εκ τούτου, πρέπει να υποτεθεί ότι τα παραπάνω αποτελέσματα επηρεάζονται από αυτές τις συσχετίσεις μεταξύ του εκπαιδευτικού και του επικυρωτικού συνόλου, και να πραγματοποιηθεί η σύγκριση σε νέα δεδομένα.

Για να το επιτύχουμε αυτό, η εκπαίδευση πραγματοποιήθηκε ξανά, αποκλείοντας 15 από τα 56 αρχεία δεδομένων, τα οποία χρησιμοποιήθηκαν μόνο για την εξαγωγή συμπερασμάτων ώστε να μετρηθεί το πραγματικό MSE του μοντέλου. Ο μέσος όρος του MSE υπολογίστηκε επίσης για τα αποτελέσματα που παράγονται από απλή επέκταση της τροχιάς στα ίδια δεδομένα. Όπως αναμενόταν, το MSE για τα νέα δεδομένα είναι 20.5, πολύ υψηλότερο από την απώλεια που μετρήθηκε νωρίτερα στο σύνολο επικύρωσης. Για τις προβλέψεις της απλής επέκτασης, το MSE είναι σχεδόν 25, που είναι σημαντικά υψηλότερο.

6.3.3 Αποτελέσματα

Με βάση τα παραπάνω πειράματα και παρατηρήσεις, το μοντέλο που τελικά επιλέχθηκε είναι αυτό που εκπαιδεύτηκε με batch_size=32, φθίνοντα ρυθμό εκμάθησης και 4 πλησιέστερους γεί-

τονες χωρίς περιορισμούς απόστασης. Τα αποτελέσματα δεν μπορούν εύκολα να αξιολογηθούν οπτικά, καθώς, όπως αναφέρθηκε, η προβλεπόμενη τροχιά λαμβάνει υπόψη την εκτιμώμενη μελλοντική κίνηση του UAV. Ορισμένα παραδείγματα μπορούν να φανούν στις παρακάτω εικόνες:

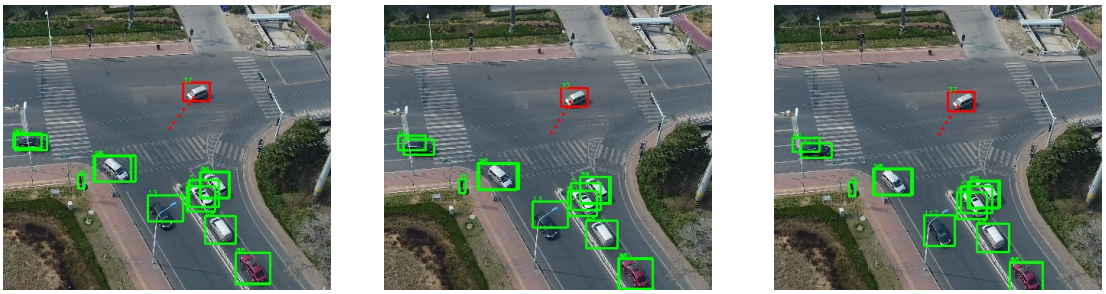


Figure 6.11: Παράδειγμα περίπτωσης όπου η προβλεπόμενη διαδρομή μέσα στο καρέ ακολουθεί την πραγματική διαδρομή.

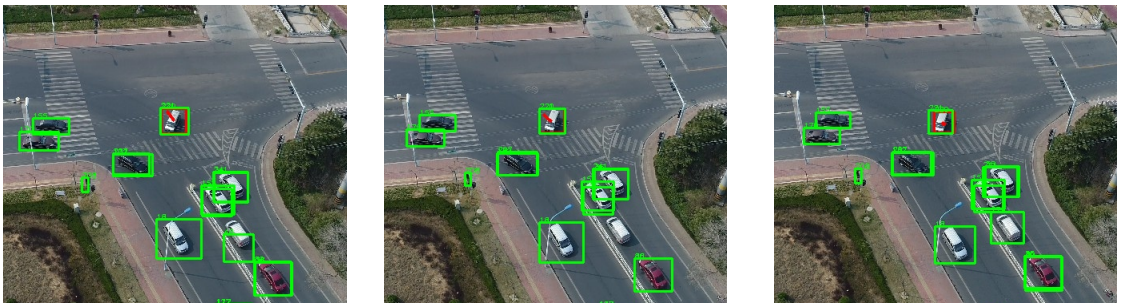


Figure 6.12: Παράδειγμα πρόβλεψης με κίνηση της κάμερας που ακολουθεί την πορεία του οχήματος (ελαφρώς γρηγορότερα), κάνοντάς την προβλεπόμενη θέση του οχήματος να μην αντιστοιχεί στα σχετικά πραγματικά σημεία στο τρέχον καρέ.

6.4 Tracking Αντικειμένων

Ένας ανιχνευτής πολλαπλών αντικειμένων που μπορεί να επιτύχει τους στόχους που έχουν τεθεί στο Κεφάλαιο 3 είναι ο Ανιχνευτής Deep SORT, που έχει υλοποιηθεί στο πακέτο python deep-sort-realtime. Ωστόσο, κατά τη διάρκεια της δοκιμής του, διαπιστώθηκε ότι μπορεί να χρειάζεται από 2-3 μέχρι μερικές εκατοντάδες millisecond για να παραγάγει αποτελέσματα, με την καθυστέρηση να είναι προβληματική κυρίως στην περίπτωση πολλαπλών διαδρομών, όπως αναμενόταν.

Για να επιλυθεί αυτό, δεδομένου ότι η κύρια ανησυχία είναι η παρακολούθηση του επιλεγμένου στόχου, και το σύστημα μπορεί να βασιστεί μόνο στην ανίχνευση για τα υπόλοιπα οχήματα, μία εναλλακτική θα ήταν να χρησιμοποιηθεί ο ανιχνευτής CSRT που έχει υλοποιηθεί στην opencv, ο οποίος εξετάστηκε στο κεφάλαιο 2. Φυσικά, δεν παρέχει την αντιστοίχιση "βαθιών" χαρακτηριστικών του Deep SORT.

Δεδομένου ότι ο διακομιστής δεν θα δέχεται κάθε καρέ, ειδικά στην περίπτωση επικοινωνίας 4G, αυτή η επιλογή είναι ιδιαίτερα αναξιόπιστη. Για να αποφευχθεί αυτό, και επίσης να μειωθούν οι χρόνοι μετάδοσης, όπως αναφέρθηκε στο προηγούμενο κεφάλαιο, οι εικόνες μπορούν να κοπούν γύρω από τον στόχο, σε ένα τετράγωνο με μήκος πλευράς 4·max(target_width, target_length). Αυτό επιτρέπει στον διακομιστή να έχει συνήθως το πολύ 5-7 αντικείμενα για παρακολούθηση, ενώ αποκτά επίσης πληροφορίες σχετικά με την τοποθεσία των πιο σημαντικών γειτόνων για τη διαδικασία πρόβλεψης τροχιάς. Ο ανιχνευτής το διαχειρίζεται επιτυχώς, επιτυγχάνοντας σταθερά χρόνους μικρότερους των 5 ms.

6.5 Συνδυασμός Αποτελεσμάτων-Επαναεντοπισμός Στόχου

Το πρώτο βήμα για τη διαδικασία επαναεντοπισμού είναι να εκτιμηθεί πόσο πιθανό είναι για κάθε ανιχνευμένο αντικείμενο να είναι ο στόχος, με βάση τη θέση που προβλέπεται για το τρέχον καρέ από το μοντέλο πρόβλεψης τροχιάς. Αυτό μπορεί να γίνει δημιουργώντας μια κανονική 2D κατανομή πιθανότητας κεντραρισμένη γύρω από κάθε σημείο της προβλεπόμενης τροχιάς και χρησιμοποιώντας την κατάλληλη για κάθε περίπτωση, ανάλογα με το πόσα βήματα πριν έγινε η πρόβλεψη. Για να γίνει αυτό, πρέπει πρώτα να εκτιμηθεί ο πίνακας συνδιακύμανσης της κατανομής:

$$\sigma_{xx} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)^2 \tag{6.1}$$

$$\sigma_{yy} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mu_y)^2 \tag{6.2}$$

$$\sigma_{xy} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{6.3}$$

Το σκορ "κοντινότητας" του αντικειμένου σε απόσταση (x,y) από το προβλεπόμενο σημείο μπορεί να εκτιμηθεί ως εξής:

$$S_{dist} = \exp\left(-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x}{\sigma_x}\right)^2 - 2\rho\left(\frac{x}{\sigma_x}\right)\left(\frac{y}{\sigma_y}\right) + \left(\frac{y}{\sigma_y}\right)^2\right]\right) \tag{6.4}$$

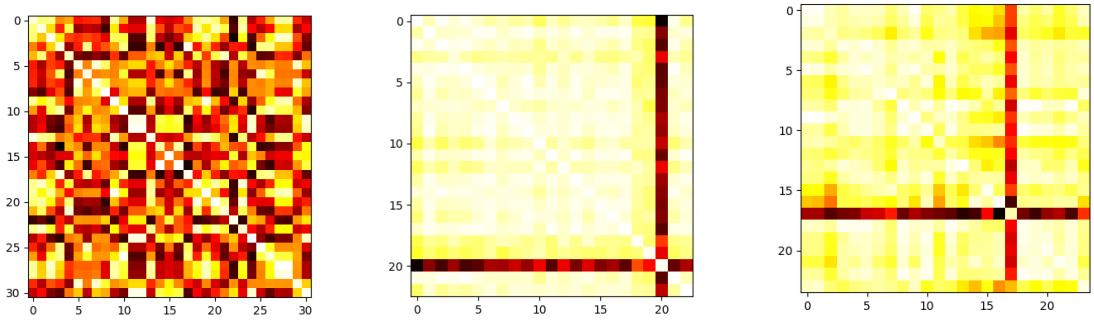
Ο παράγοντας $\frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}}$ έχει αφαιρεθεί από τον τύπο της κατανομής, έτσι ώστε η μέγιστη τιμή να είναι 1. Οι μέσες τιμές δεν περιλαμβάνονται παραπάνω, αφού το x και το y είναι αποστάσεις από το σημείο ανίχνευσης, το οποίο είναι το κέντρο της κατανομής.

Χρησιμοποιώντας τις προβλέψεις από το μοντέλο και τις αποστάσεις των προβλεπόμενων σημείων από τα πραγματικά μελλοντικά, αυτά τα αποτελέσματα υπολογίστηκαν για εικόνες διαστάσεων 1024x540:

	σ_{xx}	σ_{yy}	σ_{xy}
step 1	57.5	15.4	4.0
step 2	173.1	60.7	12.6
step 3	359.5	136.8	25.1
step 4	621.9	240.0	39.8
step 5	965.7	371.6	56.0
step 6	1422.8	538.6	67.8

Το επόμενο βήμα είναι η εκτίμηση του πόσο "μοιάζει" το κάθε αντικείμενο που έχει ανιχνευθεί με τον στόχο. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας τα χαρακτηριστικά που εξάγονται από τα μοντέλα ανίχνευσης, με την ομοιότητα συνημιτόνου μεταξύ του διανύσματος χαρακτηριστικών του εξεταζόμενου αντικειμένου και του διανύσματος χαρακτηριστικών του στόχου όταν παρατηρήθηκε τελευταία φορά ως μετρική.

Η ομοιότητα αυτή αποτελεί ένα (όχι καθοριστικό) μέτρο σύγκρισης, καθώς αυτή είναι η πληροφορία με βάση την οποία αξιολογούνται οι περιοχές ενδιαφέροντος από τα μοντέλα. Για να το δείξουμε αυτό, κατασκευάστηκαν τα παρακάτω heatmaps από τα χαρακτηριστικά που εξήχθησαν για ορισμένα σύνολα ανιχνευμένων οχημάτων:



(a) DETR (b) RT-DETR (c) YOLOv8x

Figure 6.13: Το στοιχείο (i,j) του heatmap υποδεικνύει την ομοιότητα συνημιτόνου μεταξύ του διανύσματος χαρακτηριστικών της διαδρομής j στο καρέ 0 και του μέσου διανύσματος χαρακτηριστικών της διαδρομής i στα επόμενα 3 καρέ. Υψηλότερες τιμές παρατηρούνται στις διαγώνιες.

Ως τελευταίο βήμα, η ομοιότητα των διαστάσεων του bounding box μπορεί να εκτιμηθεί ως

$$S_{dim} = 1 - \frac{|\Delta w|}{2 \cdot max(w)} - \frac{|\Delta h|}{2 \cdot max(h)} \tag{6.5}$$

όπου $\Delta w, \Delta h$ είναι η διαφορά πλάτους και ύψους αντίστοιχα και $max(w), max(h)$ είναι το μεγαλύτερο πλάτος και ύψος. Τέλος, η συνολική ομοιότητα μπορεί να εκφραστεί ως ένα σταθμισμένο άθροισμα των παραπάνω:

$$S_{tot} = W_{dist} \cdot S_{dist} + W_{feat} \cdot S_{feat} + W_{dim} \cdot S_{dim} \tag{6.6}$$

Βάσει της σημασίας κάθε μιας απο τις μετρικές, τα παραπάνω βάρη ορίστηκαν σε $W_{dist} = 0.6$, $W_{feat} = 0.3$ and $W_{dim} = 0.1$.

Κεφάλαιο 7

Συμπεράσματα

Η συζητηθείσα αρχιτεκτονική μπορεί να προσφέρει σημαντική βοήθεια στην καθοδήγηση ενός UAV, συνδυάζοντας την τοπική επεξεργασία χαμηλών δυνατοτήτων και την πιο σύνθετη απομακρυσμένη ανάλυση της κατάστασης. Η απλή διαδικασία ανίχνευσης και παρακολούθησης που εκτελείται στο UAV του επιτρέπει να πλοηγείται αυτόματα, ενώ ένας απομακρυσμένος server υψηλών επιδόσεων παρέχει οδηγίες που μπορούν να αποτρέψουν ή να διορθώσουν λανθασμένες αποφάσεις, ειδικά σε πιο σύνθετες καταστάσεις.

Ένα Raspberry Pi χρησιμοποιήθηκε για την επεξεργασία στο UAV, μαζί με μια συσκευή NCS2 που επιτρέπει την εκτέλεση μοντέλων ανίχνευσης αντικειμένων, δεδομένων των περιορισμένων διαθέσιμων υπολογιστικών πόρων. Αυτή η τοπική επεξεργασία περιλαμβάνει την ανίχνευση αντικειμένων ενδιαφέροντος με ένα μοντέλο YOLOv5n εκπαιδευμένο στο σύνολο δεδομένων VisDrone2018-DET και την παρακολούθηση ενός ορισμένου στόχου χρησιμοποιώντας έναν ανιχνευτή της OpenCV. Το τοπικό σύστημα είναι επίσης υπεύθυνο για την κατάλληλη καθοδήγηση του UAV, στέλνοντας τις απαραίτητες εντολές Mavlink στον αυτόματο πιλότο. Το Raspberry Pi προσαρμόζει την εκτίμησή του σχετικά με τον στόχο με βάση τις κατευθύνσεις που δίνονται από τον απομακρυσμένο server, υπό την προϋπόθεση ότι υπάρχει αξιόπιστη σύνδεση μεταξύ τους.

Ο server εκτελεί μια πιο ακριβή και αξιόπιστη ανίχνευση αντικειμένων, χρησιμοποιώντας ένα μοντέλο YOLOv8x, επίσης εκπαιδευμένο στο αναφερόμενο σύνολο δεδομένων. Είναι επίσης υπεύθυνος για την πρόβλεψη της μελλοντικής τροχιάς του στόχου και τον επαναεντοπισμό του όταν είναι απαραίτητο, χρησιμοποιώντας έναν συνδυασμό αυτής της πρόβλεψης τροχιάς και διανυσμάτων χαρακτηριστικών που εξάγει από τα ανιχνευμένα αντικείμενα, ενώ παρέχει βοηθητικές κατευθύνσεις σχετικά με την τοποθεσία του στόχου στο UAV.

7.1 Μελλοντικές Κατευθύνσεις

Ένα σημαντικό ζήτημα που μπορεί να επιλυθεί σε πιθανές μελλοντικές επεκτάσεις της παραπάνω εργασίας είναι η βελτίωση πρακτικών ζητημάτων που εμποδίζουν τη σωστή λειτουργία του συστήματος. Ένα τέτοιο ζήτημα είναι η εξαιρετικά αργή μετάδοση δεδομένων χρησιμοποιώντας το 4G HAT. Η χρήση διαφορετικού υλικού θα μπορούσε πιθανώς να βελτιώσει το συγκεκριμένο

πρόβλημα. Η σύνδεση σε δίκτυο 5G θα μπορούσε επίσης να φέρει μια αξιοσημείωτη διαφορά, εφόσον υπάρχει ικανοποιητική ποιότητα σήματος. Ένα άλλο πιθανό βήμα είναι η αντικατάσταση του NCS2 με μια πιο πρόσφατη μονάδα παρόμοιας λειτουργίας, που θα μπορούσε να επιτρέψει την εκτέλεση νεότερων και πιθανώς πιο σύνθετων μοντέλων στο Raspberry Pi, παράγοντας καλύτερα αποτελέσματα και αποφεύγοντας ζητήματα συμβατότητας.

Σημαντικά βήματα μπορούν επίσης να γίνουν σχετικά με τους στόχους του συστήματος. Αντί για το tracking και την ακολούθηση στόχων, πιο σύνθετες αποστολές μπορούν να του ανατεθούν, καθώς μπορεί να το επιτρέψει η υπολογιστική ισχύς του server. Τέτοιες αποστολές θα απαιτούσαν κάποια μορφή σημασιολογικής ανάλυσης των σκηνών, που θα μπορούσε επίσης να επιτρέψει αξιόπιστες προβλέψεις σχετικά με την μελλοντική τους εξέλιξη, οδηγώντας σε καλύτερη σύνθετη λήψη αποφάσεων.

Αυτό θα μπορούσε να επιτευχθεί με τη δημιουργία μιας αναπαράστασης του περιβάλλοντος του UAV (χωρική και χρονική) και των σχέσεων μεταξύ των στοιχείων του, πιθανώς χρησιμοποιώντας πανοπτική κατάτμηση (panoptic segmentation), σε συνδυασμό με πρόσθετες πληροφορίες όπως η εκτίμηση τοποθεσίας ή βάθους. Αυτό θα ενίσχυε όχι μόνο την "κατανόηση" της τρέχουσας κατάστασης, αλλά και τις μελλοντικές εκτιμήσεις. Μια τέτοια αναπαράσταση θα μπορούσε να αποκτηθεί με τη μορφή γράφου, χρησιμοποιώντας μια μέθοδο παρόμοια με αυτή που περιγράφεται στο [6]. Οι απαραίτητες πληροφορίες μπορούν επίσης να παρέχονται από πολλαπλά UAV και να συνδυάζονται, οδηγώντας σε πιο ακριβή αποτελέσματα.

Βιβλιογραφία

[1] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer Normalization". In: *ArXiv* abs/1607.06450 (2016). URL: https://api.semanticscholar.org/CorpusID:8236317.

[2] Alex Bewley et al. "Simple online and realtime tracking". In: *2016 IEEE International Conference on Image Processing (ICIP)* (2016), pp. 3464–3468. URL: https://api.semanticscholar.org/CorpusID:16034699.

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *ArXiv* abs/2004.10934 (2020). URL: https://api.semanticscholar.org/CorpusID:216080778.

[4] David S. Bolme et al. "Visual object tracking using adaptive correlation filters". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 2544–2550. DOI: 10.1109/CVPR.2010.5539960.

[5] Nicolas Carion et al. "End-to-End Object Detection with Transformers". In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Cham: Springer International Publishing, 2020, pp. 213–229. ISBN: 978-3-030-58452-8.

[6] Y. Cong et al. "Spatial-Temporal Transformer for Dynamic Scene Graph Generation". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2021, pp. 16352–16362. DOI: 10.1109/ICCV48922.2021.01606. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.01606.

[7] Corinna Cortes and Vladimir Naumovich Vapnik. "Support-Vector Networks". In: *Machine Learning* 20 (1995), pp. 273–297. URL: https://api.semanticscholar.org/CorpusID:52874011.

[8] N. Dalal and B. Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[9] Zheng Ge et al. "YOLOX: Exceeding YOLO Series in 2021". In: *ArXiv* abs/2107.08430 (2021). URL: https://api.semanticscholar.org/CorpusID:236088010.

[10] Ross Girshick. "Fast R-CNN". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[11] Ross Girshick et al. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81.

[12] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[13] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[14] Kaiming He et al. “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37 (June 2014). DOI: 10.1109/TPAMI.2015.2389824.

[15] David Held, Sebastian Thrun, and Silvio Savarese. “Learning to Track at 100 FPS with Deep Regression Networks”. In: *European Conference Computer Vision (ECCV)*. 2016.

[16] J. F. Henriques et al. “High-Speed Tracking with Kernelized Correlation Filters”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 37.03 (Mar. 2015), pp. 583–596. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2014.2345390.

[17] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[18] Yangqing Jia et al. “Caffe: Convolutional Architecture for Fast Feature Embedding”. In: *Proceedings of the 22nd ACM international conference on Multimedia* (2014). URL: https://api.semanticscholar.org/CorpusID:1799558.

[19] Glenn Jocher. *Ultralytics YOLOv5*. Version 7.0. 2020. DOI: 10.5281/zenodo.3908559. URL: https://github.com/ultralytics/yolov5.

[20] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. Version 8.0.0. 2023. URL: https://github.com/ultralytics/ultralytics.

[21] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. “Forward-Backward Error: Automatic Detection of Tracking Failures”. In: *2010 20th International Conference on Pattern Recognition*. 2010, pp. 2756–2759. DOI: 10.1109/ICPR.2010.675.

[22] Harold W. Kuhn. “The Hungarian method for the assignment problem”. In: *Naval Research Logistics (NRL)* 52 (1955). URL: https://api.semanticscholar.org/CorpusID:9426884.

[23] Chuyin Li et al. “YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications”. In: *ArXiv* abs/2209.02976 (2022). URL: https://api.semanticscholar.org/CorpusID:252110986.

[24] T. Lin et al. “Feature Pyramid Networks for Object Detection”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, July 2017, pp. 936–944. DOI: 10.1109/CVPR.2017.106. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.106.

[25] Tsung-Yi Lin et al. “Focal Loss for Dense Object Detection”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[26] Tsung-Yi Lin et al. “Microsoft COCO: Common Objects in Context”. In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

[27] Shu Liu et al. “Path Aggregation Network for Instance Segmentation”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 8759–8768. URL: https://api.semanticscholar.org/CorpusID:3698141.

[28] Bruce D. Lucas and Takeo Kanade. “An iterative image registration technique with an application to stereo vision”. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI’81. Vancouver, BC, Canada: Morgan Kaufmann Publishers Inc., 1981, pp. 674–679.

[29] Alan Lukežic et al. “Discriminative Correlation Filter with Channel and Spatial Reliability”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4847–4856. DOI: 10.1109/CVPR.2017.515.

[30] Wenyu Lv et al. “DETRs Beat YOLOs on Real-time Object Detection”. In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023), pp. 16965–16974. URL: https://api.semanticscholar.org/CorpusID:258179840.

[31] Joseph Redmon and Ali Farhadi. “YOLO9000: Better, Faster, Stronger”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 6517–6525. URL: https://api.semanticscholar.org/CorpusID:786357.

[32] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *ArXiv* abs/1804.02767 (2018). URL: https://api.semanticscholar.org/CorpusID:4714433.

[33] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[34] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (June 2015). DOI: 10.1109/TPAMI.2016.2577031.

[35] B. Scholkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning series. MIT Press, 2018. ISBN: 9780262536578. URL: https://books.google.gr/books?id=ZQxiuAEACAAJ.

[36] Jasper Uijlings et al. “Selective Search for Object Recognition”. In: *International Journal of Computer Vision* 104 (Sept. 2013), pp. 154–171. DOI: 10.1007/s11263-013-0620-5.

[37] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[38] P. Viola and M. Jones. “Rapid object detection using a boosted cascade of simple features”. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 1. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.

[39] Ao Wang et al. “YOLOv10: Real-Time End-to-End Object Detection”. In: *arXiv preprint arXiv:2405.14458* (2024).

[40] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. “YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors”. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 7464–7475. DOI: 10.1109/CVPR52729.2023.00721.

[41] Chien-Yao Wang and Hong-Yuan Mark Liao. “YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information”. In: *arXiv preprint arXiv:2402.13616* (2024).

[42] Chien-Yao Wang et al. “CSPNet: A New Backbone that can Enhance Learning Capability of CNN”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2019), pp. 1571–1580. URL: https://api.semanticscholar.org/CorpusID:208310312.

[43] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. “Simple online and realtime tracking with a deep association metric”. In: *2017 IEEE International Conference on Image Processing (ICIP)*. 2017, pp. 3645–3649. DOI: 10.1109/ICIP.2017.8296962.

[44] Pengfei Zhu et al. “VisDrone-DET2018: The Vision Meets Drone Object Detection in Image Challenge Results”. In: *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part V*. Munich, Germany: Springer-Verlag, 2019, pp. 437–468. ISBN: 978-3-030-11020-8. DOI: 10.1007/978-3-030-11021-5_27. URL: https://doi.org/10.1007/978-3-030-11021-5_27.

[45] Pengfei Zhu et al. “VisDrone-VDT2018: The Vision Meets Drone Video Detection and Tracking Challenge Results”. In: *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part V*. Munich, Germany: Springer-Verlag, 2019, pp. 496–518. ISBN: 978-3-030-11020-8. DOI: 10.1007/978-3-030-11021-5_29. URL: https://doi.org/10.1007/978-3-030-11021-5_29.

[46] Xizhou Zhu et al. “Deformable DETR: Deformable Transformers for End-to-End Object Detection”. In: *ArXiv* abs/2010.04159 (2020). URL: https://api.semanticscholar.org/CorpusID:222208633.